Rec'd PCT/PTO 03 MAY 2005
PCT/AU03/01473

#2

REC'D 01 DEC 2003
WIPO PCT

**Patent Office**
**Canberra**

I, JANENE PEISKER, TEAM LEADER EXAMINATION SUPPORT AND SALES hereby certify that annexed is a true copy of the Provisional specification in connection with Application No. 2003901924 for a patent by NOEL WILLIAM LOVISA as filed on 22 April 2003.

WITNESS my hand this
Twentieth day of November 2003

J.K.+C.

JANENE PEISKER
TEAM LEADER EXAMINATION
SUPPORT AND SALES

**PRIORITY DOCUMENT**
SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1(a) OR (b)

BEST AVAILABLE COPY

Our Ref: 7762430

P/00/009
Regulation 3:2

AUSTRALIA

Patents Act 1990

# PROVISIONAL SPECIFICATION


Applicant(s): Noel William Lovisa
2 Dianella Court
Annandale
Townsville Queensland 4814
Australia

Address for Service: DAVIES COLLISON CAVE
Patent & Trade Mark Attorneys
Level 10, 10 Barrack Street
SYDNEY NSW 2000


Invention Title: **Service implementation**

The invention is described in the following statement:

# SERVICE IMPLEMENTATION

## Background of the Invention

The present invention relates to a method and apparatus for performing a service, and in particular, to performing a service using components, each of which corresponds to a respective component service.

## Description of the Prior Art

The reference to any prior art in this specification is not, and should not be taken as, an acknowledgement or any form of suggestion that the prior art forms part of the common general knowledge.

Currently most industries utilise a degree of specialisation. However, the degree to which this is achieved, particularly in the service industries, such as accountancy, the legal profession or the like, is limited.

For example, when obtaining accountancy services, it is usual for a customer to establish a relationship with a specific accountant company or individual that can provide all the services required by the customer. This is achieved by having the accountant provide a wide range of services that will allow the accountant to obtain sufficient customers to support the practice. Thus, larger accountancy firms will generally provide a greater range of services than smaller firms, to allow the larger firms to support a larger customer base, and hence make a greater profit.

In general, when a customer wishes to obtain accountancy services, the customer will examine the services provided by a range of different firms and/or individuals, and then select a respective one to provide all their accountancy needs.

However, this suffers from a number of drawbacks. In particular, it is necessary for the accountancy firms to have expertise in a wide range of areas in order to be able to compete within the market place. This requires firms to therefore generalise their practice.

In a large accountancy firm, this is overcome to a certain extent by having individuals within the firm practice in specific areas, with work being transferred internally within the firm to allow a range of services to be performed. Alternatively, in the case of smaller firms, outsourcing can be performed, to allow specific tasks to be performed by more experienced contractors, or the like.

However, this procedure suffers from a number of drawbacks. Primarily, this requires that individuals working within the firms are capable of a large degree of generalisation, with many individuals in the service area each attempting to provide many often disparate services. As a result, there is little opportunity for any one individual within the industry to specialise on providing a narrow range of well defined services.

This lack of specialisation in turn leads to inefficiencies within the service industry. In particular, the service industries often suffer from complexity limits due to the fact that as the level of complexity rises, it becomes impossible for one person or entity to understand all aspects of a particular service requirement. Furthermore, generalists will typically perform tasks far less efficiently than specialists, not least because they are less familiar with the processes needed to perform the tasks.

Furthermore, the manner in which defined tasks are implemented is usually invisible to the customer, who is therefore often unaware of who is performing specific work on a project or the like. This removes control from the customer, preventing the customer

from specifying the individuals that should be performing respective portions of the defined task.

It will be appreciated that this applies to almost any service industry to a greater or lesser degree, but is most pronounced in service industries that involve the manipulation of data or other information.

**Summary of the Present Invention**

In a first broad form the present invention provides a method of allowing a user to obtain a service using a base station, the method utilising components, each component representing a respective service portion provided by a respective entity, the method including causing the base station to:

a) Provide an indication of one or more available components to the user;
b) Determine a combination of selected components in accordance with input commands received from the user; and,
c) Implement the selected components in accordance with the component combination, thereby causing the desired service to be performed.

The base station is usually coupled to one or more end stations via a communications network, the method including allowing the user to use the end station to:

a) Select one or more of the components; and,
b) Define a component combination using the selected components.

The method typically includes causing the base station to:

a) Receive a component request from the end station;
b) Transfer an indication of one or more components to the end station in accordance with the request, thereby allowing the user to select one or more of the components.

The method may include causing the base station to:

a) Receive a component selection from the end station, the component selection indicating one or more selected components; and,

b) Transfer details of the selected components to the end station in response to the request, thereby allowing the user to define the component combination.

The base station can include a store for storing component specifications representing the service portion provided by a respective component and, a processor, in which case the method can include causing the processor to:

a) Access the component specifications stored in the store; and,

b) Provide an indication of the services provided by the components to the end station, thereby allowing the user to select respective ones of the components.

At least some of the components typically include one or more inputs and one or more outputs, the store being further adapted to store:

a) Input specifications, each input specification indicating any information required at a respective input to perform the respective service portion; and,

b) Output specifications, each output specification indicating any information provided at a respective output, the method including causing the processor to:

i) Access the input and/or output specifications stored in the store; and,

ii) Provide an indication of the information to the end station, thereby allowing the user to select the components.

The method can include allowing the user to define the component combination by connecting the outputs and inputs of the selected components using the end station.

The method may include connecting an input and output in accordance with the information provided at the output and the information required by the input.

The method typically includes causing the base station and/or the end station to:

a) Generate a graphical representation of the selected components; and,

b) Manipulate the graphical representation in response to input commands received from the user to thereby define the component combination.

The method typically further includes causing the base station to:

a) Obtain a graphical representation of the selected components;

b) Transfer the graphical representations to the end station.

The method may include causing the base station and/or the end station to implement the combined components in accordance with the generated graphical representation.

The method preferably includes causing each of the entities to implement the respective service portions in turn to thereby provide the service.

The method of implementing the combined components can include:

a) Causing the base station to transfer an implementation order to the respective entity; and,

b) Causing the entity to provide the respective service portion in response to the received implementation order.

Each entity usually has one or more respective entity stations coupled to the base station via a communications network, the method including transferring the implementation order to the entity station, the entity station being responsive to the implementation order to cause the respective service portion to be performed.

At least some of the components include one or more inputs adapted to receive predetermined information and one or more outputs adapted to provide predetermined information, the method preferably include:

a) Causing each entity station to implement an agent associated with each input and output;

b) Causing the agents associated with an input and output to be connected to:

i) Cooperate to compare the predetermined information provided at the output and the predetermined information provided at the input; and,

ii) Determine the input and output can be connected in response to a successful comparison.

The entity stations can be adapted to cause the respective service portion to be implemented in response to a determination that each input and output associated with the component can be connected.

At least some of the services may be adapted to manipulate information, the entity being adapted to perform the service by:

a) Receiving the information to be manipulated at an input;

b) Perform the manipulation; and,

c) Provide the manipulated information at an output.

The method may include transferring the manipulated information to one or more components in accordance with the defined component combination.

The method can include causing the base station to:

a) Determine a performance information, the performance information being representative of one or more criteria regarding the implementation of the components by the respective entities;

b) Provide the performance information to the user, the user selecting the components in accordance with the performance information.

The performance information can include at least one of:

a) An indication of the entity implementing the component;
b) An indication of the geographical location of the entity;
c) An indication of the duration for implementing the component;
d) An indication of a cost associated with implementing the respective component; and,
e) A rating, the rating being indicative of the success of the component.

The method may include:

a) Providing a number of different components for performing equivalent service portions, the different components being provided by different entities; and,
b) Inducing competition between the entities to thereby drive improvement of the components.

The method typically includes generating revenue by charging a fee for the use of each component.

The method can include:

a) Providing at least some of the revenue to the entity implementing the respective component; and,
b) Having an operator of the base station retain at least some of the revenue.

In a second broad form the present invention provides a system for allowing a user to obtain a service, the base station being adapted to utilise components, each component

representing a respective service portion provided by a respective entity, the system including a base station adapted to:

a) Provide an indication of one or more available components to the user;

b) Determine a combination of selected components in accordance with input commands received from the user; and,

c) Implement the selected components in accordance with the component combination, thereby causing the desired service to be performed.

The base station can be coupled to one or more end stations via a communications network, the end station being adapted to allow the user to:

a) Select one or more of the components; and,

b) Define a component combination using the selected components.

The base station may be adapted to:

a) Receive a component request from the end station;

b) Transfer an indication of one or more components to the end station in accordance with the request, thereby allowing the user to select one or more of the components.

The base station can be adapted to:

a) Receive a component selection from the end station, the component selection indicating one or more selected components; and,

b) Transfer details of the selected components to the end station in response to the request, thereby allowing the user to define the component combination.

The base station may include a store for storing component specifications representing the service portion provided by a respective component and, a processor, the processor being adapted to:

a) Access the component specifications stored in the store; and,

b) Provide an indication of the services provided by the components to the end station, thereby allowing the user to select respective ones of the components.

At least some of the components include one or more inputs and one or more outputs, in which case the store is preferably adapted to store:

a) Input specifications, each input specification indicating any information required at a respective input to perform the respective service portion; and,

b) Output specifications, each output specification indicating any information provided at a respective output, the processor being adapted to:

i) Access the input and/or output specifications stored in the store; and,

ii) Provide an indication of the information to the end station, thereby allowing the user to select the components.

The end station may be adapted to allow the user to define the component combination by connecting the outputs and inputs of the selected components.

The input and output can be connected in accordance with the information provided at the output and the information required by the input.

At least one of the base station and the end station are generally adapted to:

a) Generate a graphical representation of the selected components; and,

b) Manipulate the graphical representation in response to input commands received from the user to thereby define the component combination.

The base station can be adapted to:

a) Obtain a graphical representation of the selected components;

b) Transfer the graphical representations to the end station.

The base station may be adapted to implement the combined components in accordance with the generated graphical representation.

The base station can include a store for storing graphical representations of each of the components, and a processor, the processor being adapted to transfer a graphical representation for each of the selected components to the end station, thereby allowing the user to manipulate the graphical representations using the end station.

Each of the entities may be adapted to implement the respective service portions in turn to thereby provide the service.

The base station can be adapted to transfer an implementation order to the respective entity, the entity responding to the implementation order to provide the respective service.

Each entity can have one or more respective entity stations coupled to the base station via a communications network, the base station being adapted to transfer the implementation order to the entity station, the entity station being responsive to the implementation order to cause the respective service portion to be performed.

At least some of the components generally include one or more inputs adapted to receive predetermined information and one or more outputs adapted to provide predetermined information, each entity station being adapted to implement an agent associated with each input and output, the agents associated with an input and output to be connected being adapted to:

a) Cooperate to compare the predetermined information provided at the output and the predetermined information provided at the input; and,

b) Determine the input and output can be connected in response to a successful comparison.

The entity stations typically are adapted to cause the respective service portion to be implemented in response to a determination that each input and output associated with the component can be connected.

At least some of the services may be adapted to manipulate information, the entity being adapted to perform the service by:

a) Receiving the information to be manipulated at an input;
b) Perform the manipulation; and,
c) Provide the manipulated information at an output.

The entity can be adapted to transfer the manipulated information to one or more components in accordance with the defined component combination.

The base station may be adapted to:

a) Determine a performance information, the performance information being representative of one or more criteria regarding the implementation of the components by the respective entities;
b) Provide the performance information to the user, the user selecting the components in accordance with the performance information.

The performance information can include at least one of:

a) An indication of the entity implementing the component;
b) An indication of the geographical location of the entity;
c) An indication of the duration for implementing the component;

d) An indication of a cost associated with implementing the respective component; and,

e) A rating, the rating being indicative of the success of the component.

The base station may be adapted to:

a) Provide a number of different components for performing equivalent service portions, the different components being provided by different entities; and,

b) Induce competition between the entities to thereby drive improvement of the components.

The base station can be adapted to generate revenue by charging a fee for the use of each component.

The base station may be adapted to:

a) Provide at least some of the revenue to the entity implementing the respective component; and,

b) Retain at least some of the revenue.

**Brief Description of the Drawings**

An example of the present invention will now be described with reference to the accompanying drawings, in which: -

Figure 1 is a flow diagram outlining an example of the provision of a service in accordance with the present invention;

Figure 2 is a schematic diagram of an example of a web based system for providing services;

Figure 3 is a schematic diagram of an example of a processing system of a base station of Figure 2;

Figure 4 is a schematic diagram of an example of an end station of Figure 2;

Figure 5 is a schematic diagram of an example of an entity processing system of Figure 2;

Figure 6 is a flow diagram of an example of the method of having an entity provide a component to the base station of Figure 2;

Figure 7 is a schematic diagram of an example of a component properties dialog box;

Figure 8 is a schematic diagram of an example of an output properties dialog box;

Figures 9A to 9E are a flow diagram of an example of the method of performing a service using the system of Figure 2;

Figure 10 is an example of a schematic representation that is presented to the user;

Figure 11 is an example of a component representation that is presented to the user;

Figure 12 is an example of the schematic representation of Figure 11 modified to include an additional component;

Figure 13 is an example of the schematic representation of Figure 12 modified to include interconnections;

Figure 14 is an example of a schematic representation of the internal structure of the component of Figure 12;

Figure 15 is a schematic diagram of an example of a schematic representation for two interconnected components;

Figure 16 is a schematic diagram demonstrating the operation of the agents of Figure 15;

Figure 17A to 17E are schematic diagrams of a first example demonstrating the operation of hand off of agents;

Figure 18 is a schematic diagram of a second example demonstrating the operation of hand off of agents;

Figures 19A and 19B are schematic diagrams demonstrating the operation of agent bundles;

Figure 20 is a schematic diagram demonstrating the operation of a sequence of agent bundles;

Figure 21 is a schematic diagram demonstrating the operation of a debundle component;

Figures 22A to 22C are schematic diagrams of a first example of bundle and debundle use;

Figure 22D is a schematic diagram of a second example of bundle and debundle use;

Figure 23 is a schematic diagram demonstrating the operation of bundle reordering;

Figure 24 is a schematic diagram demonstrating the operation of bundle rake-out; and,

Figure 25 is an example of a schematic representation for a service that allows patents to be obtained.

**Detailed Description of the Preferred Embodiments**

An example of the method by which a service may be obtained will now be described.

As shown, the first step is to determine the requirements for the service to be performed at step 100. This is achieved by considering the service required and in particular, the tasks that need to be completed by the performance of the service. For example, this may be a single basic task such as the completion of a tax return, or alternatively, may be a more complex process involving the completion of a number of tasks, such as the provision of audit, tax and other accountancy services for a firm. Similarly, the service may be the drafting of a patent application, obtaining an International Patent Portfolio, or litigating a Patent. Examples will be described in more detail below but it will be appreciated that the techniques may be applied to any service industry.

Once the desired service has been determined, a number of components are selected that when combined in an appropriate manner will allow the service to be performed. In this regard, each component corresponds to a respective service or portion of a service (hereinafter referred to generally as a component service), which is used to perform all or part of the tasks required to provide the desired overall service.

In general, the component services will involve the manipulation of some form of data or

other information, such as accounts or other financial information. Thus, for example, the component services can include processes such as the modification, creation or deletion of information, determination of data from information, or the like. This allows each component to contribute in some way to the performance of the overall service.

The component services may be performed automatically through the use of components formed from computer executable code, or the like. Typically however, the component services will be performed at least partially manually, or through combination of manual and automatic implementation.

The level of complexity of the component services will vary as will be explained in more detail below. Thus, for example, simple components may perform a single basic task, such as calculating a single value from received data, with this value being supplied to other components for subsequent processing. However, more complex components may operate to perform several tasks, such as calculating all or part of a tax return.

In order to achieve this, each component may be adapted to receive information, or other data, via one or more respective inputs, and then perform manipulations of the information as required. Similarly, the majority of components will also include one or more outputs for allowing manipulated information, data, or other information to be output.

Thus, for example, the components may be adapted to receive details of accounts from a company, such as invoice information, thereby allowing the components to determine a tax return. In this case, if the company is an international company, then information will be transferred to a number of different components to allow separate tax returns to be determined for different jurisdictions. Furthermore, whilst the information may be initially transferred to a single component, the information may then be distributed to a

number of different components to allow the tax return calculation to be performed as a number of specific tasks, with each component performing a specific task.

In use, the components interact with each other by transferring data or other information therebetween. Thus, for example, the output of one component may be connected to the input of another component, to allow two component services to be performed in sequence. Combining appropriate ones of the more basic level components in a hierarchical structure can also be used to allow more complicated services to be implemented as a collection of more basic component services.

Examples of components are set out in more detail below.

At step 120, a combination of the selected components is defined which will allow the desired service to be performed. In particular, this specifies how the components should be interconnected via the inputs and outputs, such that when the component services provided by the components are implemented at step 130, the interaction results in the performance of the service.

It will be appreciated that the process described above may be implemented using a number of different architectures. Thus, for example, the process may be implemented using a single processing system which has the ability to implement or at least co-ordinate the implementation of each of the components. However, typically the process is implemented using a distributed web based system, or the like, with user accessing facilities provided by a central base station via the Internet, or another communications network.

An example of this will now be described in more detail with respect to Figure 2.

In particular, in this example, the apparatus includes at least one base station 1 coupled via a communications network, such as the Internet 2, and/or a number of local area networks (LANs) 4, to a number of end stations 3.

In use, the base station 1 is adapted to allow users of the end stations 3 to select components from a database, define the component combination, and then implement the defined component combination in order to obtain the desired service. The selection of components will normally be achieved by selecting component representations as will be described in more detail below.

Accordingly, the base station 1 and the end stations 3 are adapted to communicate to allow details of the components to be transferred therebetween, allowing users to select components, and to determine component combinations defining the service required. This may be achieved in a number of manners, although the following examples will focus on the use of web pages.

In order to achieve this, the base station 1 typically includes one or more processing systems 10 (two shown in this example for clarity purposes only) coupled to one or more respective databases 11.

An example of a suitable processing system 10 is shown in Figure 3. As shown, the processing system 10 generally includes at least a processor 20, a memory 21, an input device 22, such as a keyboard, or the like, an output device 23, such as a monitor, which are coupled together via a bus 24. The processing system is also provided with an external interface 25 for coupling the base station 1 to the Internet 2, or the LAN 4, as well as the database 16, as required.

The processing systems 10 are adapted to generate web pages for viewing using the end

stations 3, as well as to implement component combinations defined by the end station user's. From this, it will be appreciated that the processing system 10 may be any form of processing system such as a computer, a laptop, server, specialised hardware, or the like.

Similarly, each end station 3 is adapted to communicate with the base station 1 to access web pages and transfer data to the base station 1, as required.

An example of a suitable end station 3 is shown in Figure 4. As shown the end station 3 includes a processor 30, a memory 31, an input device 32, such as a keyboard, or the like, an output device 33, such as a display, which are coupled together via a bus 34. The processing system is also provided with an external interface 35 for coupling the end station 3 to the Internet 2, or the LAN 4, as required.

In use, the processor 30 is adapted to communicate with the processing system 10 provided in the base station 1 via the communications networks 2, 4 to allow the processing system services to be accessed. Accordingly, it will be appreciated that the end stations 3 may be formed from any suitable processing system, such as a suitably programmed PC, Internet terminal, lap-top, hand-held PC, or the like, which is typically operating applications software to enable data transfer and in some cases web-browsing.

The components may be provided at, and implemented by, the processing system 10. However, typically the components are provided by one or more respective entities, each of which operates one or more respective entity stations 5, which are also coupled to the Internet 2, and/or the LANs 4, as shown. In this example, each entity station 5 is formed from an entity processing system 15, coupled to a store, such as a database 16, as shown.

The entity processing system 15 must therefore be able to communicate with the processing system 10 via the communications networks 2, 4. In order to achieve this, the

entity processing system 15 would generally be similar to the processing system shown in Figure 5.

As shown in this example, the entity processing system 15 includes a processor 40, a memory 41, an input device 42, such as a keyboard, or the like, an output device 43, such as a monitor, which are coupled together via a bus 44. The processing system is also provided with an external interface 45 for coupling the entity station 5 to the Internet 2, or the LAN 4, as well as the database 16, as required.

In use, the processing system 15 is adapted to allow the entity to perform the services encapsulated in respective components. This may be achieved by having the service performed manually by an operator, in which case the entity station 5 acts as an interface between the operator and the remainder of the system. Alternatively, the service may be implemented automatically or through a combination of automatic and manual operation, in which case the processor 40 may perform some or all of the service. Accordingly, it will be appreciated that the entity stations 5 may be formed from any suitable processing system, such as a suitably programmed PC, Internet terminal, lap-top, hand-held PC, or the like. Typically however, as the services are data intensive, the entity processing systems 15 will be formed from servers, or the like.

To allow components to be implemented by the entity stations 5, whilst still allowing users of the end stations 3 to access the components via the base station 1, it is typical for details of the components to be stored in the database 11, in the form of component specifications. The component specifications may be in any one of a number of forms, and may include graphical representations, or the like. However, in general the component specifications include at least sufficient information for a user to determine the effect of the service embodied in the component. It will be appreciated that the manner in which the service is achieved will typically be retained as secret by the

implementing entity and will not therefore be described in the specification.

The manner in which an entity may provide details of a component in the form of component specifications will now be described with reference to Figure 6.

In particular, at step 200 the entity determines a manner of providing a respective component service. This may be achieved in a number of ways and will depend on the respective component service and the manner in which the entity wishes to provide the component service.

Thus, for example, the entity may provide the component service manually by having an individual or team receive information at the entity station 5, modify or otherwise act upon the information, and then provide an output. The output may be provided to any one of a number of other entities, subsequent components, the processing system 10, the end station 3, or another one of the entity stations as will be described in more detail below with respect to step 420. Alternatively, the component service may be performed by computer executable code, executed by the entity processing system 15, in which case, the entity must first determine the necessary executable code. A combination of manual and automatic processes may also be used.

As a further option, the entity may provide a component service in the form of a compound component. In this case, the entity effectively defines a combination of previously existing components, which when combined in a predetermined manner or with skill and knowledge in a manual manner, or semi-automatic manner allows the required component service to be performed. In this case, the entity station 5 will be adapted to hand-off implementation of the components contained within the compound component to other entities, such as through other entity stations 5, and/or the base station 1, as required.

In any event at step 210 the entity defines a component encapsulating the provision of the component service using the entity station 5. In order to achieve this, the entity processing system 15 will generally be provided with applications software that aids the entity in this process. In particular, the software will prompt the entity to provide information that will be required by the processing system 10 to allow the functionality provided by the respective component to be determined by a user. Thus, for example the entity may be presented with a dialog box including fields defining the types of information that are required in order for users to determine the component service operation of the component.

In general, the required information includes at least component, input and output specifications. In particular, the component specifications are used to provide information regarding the effect of the service provided by the component, together with information regarding the component author, implementing entity, or the like. The component specification also includes sufficient information to allow the processing system 10 or the end station 3 to access the services represented by the component.

Accordingly, the component specifications typically include at least:

- Manufacturer ID – used to identify the entity providing the service
- Component ID – used to identify the respective component
- Location information – used to identify where the component is implemented
- Description – an indication of the effect of the service provided by the component

This information may be provided for example through the use of a properties dialogue box shown for example in Figure 7. The properties dialogue box will prompt the entity to provide information such as the component name, the component description, the author, the address, report number or the like.

The applications software installed on the entity processing system 15 can also be used to generate any identifiers that may be required. In particular, it is generally necessary to generate identifiers to allow both the entity, and the component to be uniquely identified.

Furthermore, an entity station 5 may be implementing the same component simultaneously for the purpose of providing a number of different overall services. Thus, for example, the entity may have to provide the same component service for a number of different users simultaneously. In this case, several different component instances are defined to exist on the entity station 5, with each component instance being applied to each respective service provision. Accordingly, in this case, it is also necessary to generate respective identifiers allowing each component instance to be uniquely identified.

In addition to this, the entity also provides input and output specifications, which are used to indicate the types and/or formats of data that can be received by the component inputs, or output from the component outputs. This is important for ensuring that components are able to communicate with each other, by transferring data or other information from the output of one component to the input of a subsequent component.

In this example, control of this communication is achieved using agents, which are software applications executed at the location at which the respective component is implemented. The agents operate to negotiate between available data types and formats specified in the input and output specifications, to allow respective components to communicate directly.

It will be appreciated that in the situation in which the components are implemented substantially manually, the interaction between components may also occur manually.

Thus, for example, individuals providing component services at two different entities may need to talk to each other to discuss a respective service implementation. In this case, the agents could be adapted to negotiate a meeting or conference time, as well as how this may be achieved (such as through the use of video or telephone conferences, or the like).

In general, the input and output specification may also include details of the manner of operation of the respective agent. Accordingly, the details may be provided through the use of a dialog box that prompts the entity for details regarding the respective input and/or output and associated agent. An example of a dialog box for an output is shown in Figure 8.

The operation of the agents will be described in more detail below. However, it will be appreciated that the entity also operates to construct agents when encapsulating the service as a component.

It is also possible for the negotiation to be performed using techniques other than agents, depending on the implementation of the invention.

Accordingly, at step 220 the entity processing system 15 operates to store the generated component, input and output specifications, and agents, typically in the database 16. Agents or definitions thereof may also be stored. The entity station 5 is then used to access the base station 1 at step 230, allowing details of the component, input and output specifications, to be transferred to the base station 1 at step 240, for storage in the database 11 at step 250.

It will be appreciated that if the component is self contained and automatically implemented, the entire component may be downloaded to the database 11, for storage

thereon, in which case there is no requirement to store any information at the entity station 5. This allows the component service to be implemented by the processing system 10 automatically. Alternatively, the component may be transferred to the end station 3 for implementation thereon. These techniques will generally result in the manner of implementation of the services to be made publicly available.

Typically however, the entity will wish to retain at least some form of control over the operation of the component for a number of reasons, in which case the component service may be implemented at the entity station 5. This is also generally required if the service implementation requires manual input from the entity, but may also be desirable for other reasons.

Thus, for example, this allows the entity to monitor use and operation of the component, as well as making it easier for the entity to adjust and/or modify the operation of the component to improve its efficiency. Furthermore, this allows the entity supplying the service to provide only an output, and not divulge the method used to implement the service. This allows the implementation of the service to be retained as a trade secret, specialised knowledge or the like.

In any event, as the system is adapted to handle a large number of components, it is generally undesirable to have all these located at the base station 1, as the database 11, and processing systems 10 would rapidly become over used.

Accordingly, the components are usually implemented at the entity stations 5, with details of the specifications being transferred to the base station 1, to allow users of the end stations 3 to select the components for use. In particular, when the users of the system select components in this fashion, it is transparent to the user whether the component itself is actually provided at the base station 1 or whether the component is

provided at an entity station 5. This is because all the specifications needed to access the entity station 5 providing the respective service are stored in the base station 1.

A detailed example of the manner in which a user uses the base station 1 to obtain a service will now be described in more detail, with respect to the flow chart set out in Figures 9A to 9E.

Accordingly, as shown at step 300 in Figure 9A the first stage is for a user to determine the requirements of the service to be performed. At step 310 the user then accesses the base station 1 using the end station 3.

At step 320 the user selects a component search using the end station 3 and this causes the processing system 10 to provide details of available components based on component specifications stored in the database 11, at step 330. In particular, the processing system will typically allow users to search through categories of components, with the categories defining different forms of functionality. This allows users to rapidly locate components that are suitable for performing required services.

In general, different categories of component will be defined for different industries and for different groups of services within an industry. This would usually be broken down in a hierarchical fashion. Thus for example a category may be provided for accounts services, with separate sub-categories for audits, tax assessments, etc.

At step 340 the user reviews the component properties and selects one or more components. This may be achieved in a number of ways, although typically the user will be presented with navigable lists that provide at least a component title and brief additional description of suitable components and the component services provided therein. The user can then select a respective one of the components allowing further

details to be provided, and ultimately, the selection to be made. This may be based on criteria such as cost, quality of service previous implementation details or the like, as described in more detail below.

Thus, for example, the user may wish to have a patent application filed, which will require at least the patent specification to be drafted and filed. In this case, a single entity may provide a respective component for providing both services. Alternatively, these separate stages may be provided by respective entities as separate components. Thus, in this case, the user can select a drafting component corresponding to the provision of drafting services, and a filing component corresponding to the provision of filing services.

In this case, the drafting component will require details of the invention as an input, and will provide the drafted specification as the output. The filing component will require a drafted specification as an input and will provide patent applications as an output. This will be described in more detail below.

The details may be provided for example through the use of the properties dialogue box similar to that shown for example in Figure 7. In this case, the details include information such as the component name, the component description, the author, the address, report number, or the like, and will be determined directly from the component specifications stored in the database 11.

At step 350 the end station 3 (or alternatively the processing system 10) stores an indication of the selected components. This may be achieved in a number of manners depending on the implementation. Thus, for example, the end station 3 typically generates component data, which is stored in the memory 31, the component data including an indication of each component selected by the user. Alternatively, however,

the processing system 10 may generate the component data and store it in the database 11.

At step 360 the end station 3 (or alternatively the processing system 10) generates a schematic representation, including representations of the components so far selected. The schematic representation is used to allow the user to define the component interconnections, as will be described in more detail below. In particular, this allows the user of the end station 3 to visualise the components and how these will need to interact with each other to perform the required service.

The schematic representation includes a representation of each of the components selected. The component representation is generally generated by the entity and transferred to the base station 1 as part of the component specifications, although any suitable technique for generating the representation may be used. When the user selects a respective component, the corresponding component representation is transferred from the base station 1 to the end station 3, and added to the schematic representation, as required.

It will therefore be appreciated that the indication of the component stored by the end station 3 may be in the form of the component representations.

An example of a schematic representation is shown in Figure 10. As shown, the schematic representation is displayed in a schematic window 50, and in this example, includes two component representations 51, 52. Each of the components has a number of inputs and outputs, shown generally at 51A, 51B, 52A, 52B, 52C.

In the example of filing a patent application, the component 51 can correspond to a drafting component, with the component 52 corresponding to a filing component. In this

example, the drafting component 51 is adapted to receive details of the invention at the input 51A, and provide a drafted specification at the output 51B. The filing component obtains the patent specification and provides details of the filed application and the associated patent specification at the outputs 52B, 52C.

If the user selects one of the respective components shown in the schematic representation, the user is presented with a single component representation, an example of which is shown in Figure 11. In particular, Figure 11 shows a component display screen 60 including a representation of the component having the input and output representations 52A, 52B, 52C presented thereon.

The component display screen also includes a number of window selection tabs 61, which allow the user to navigate between the component window 60 shown, the schematic window 50 mentioned above, and an auto select window.

An agent window 62 is also provided, which displays details of a selected input or output agent (in this example agent 52C), obtained from the respective input and/or output specification.

In use, the user can navigate around the component and schematic representations to allow various information regarding the components to be provided. Thus, for example, by selecting the component representation 52, this can allow the properties of the corresponding component to be displayed, as shown for example in Figure 7. Similarly, by selecting a respective one of the input and/or output representations, details of the respective input or output will be displayed in the agent window 62. These details will typically be provided by displaying an input or output dialog box, similar to the one shown in Figure 8, as appropriate.

In fact the information viewable by the user is typically limited to the external schematic. The graphical user interface shown in Figure 11 may also be used by the entity in defining the component representation as described above.

In any event, the user reviews the presented schematic representation and determines if further components are required at step 370. If it is determined that more components are required at step 380, the process returns to step 320 to allow the user to return to the component search tool and select more components using the end station 3. Representations of these components can then be added to the schematic representation as required, for example using drag and drop techniques.

Once the required components (or at least some of the required components) are selected, through the placement of corresponding component representations on the schematic representation, the user determines component inputs and outputs that are to be connected at step 390.

In order to ensure that the components may interact successfully, the user will typically check at this point whether the input and output that are to be connected are compatible at step 400. In particular, the user checks whether the input and output can handle any common data types and/or formats. This information can be determined by examination of the input and output details determined from the input and output specifications. Alternatively, the component may be adapted to provide the drafted specification as a Word™ document, in which case the subsequent component must also be adapted to receive Word documents.

Thus, for example, in the case of filing a patent application, the drafting component may provide copies of the drafted specification in a first language, whereas the filing component may require the specification in a second language. In this case it will be

appreciated that the components cannot be directly interconnected, as the entity providing the filing services will be unable to provide the filing services with the specification in the first language.

If the user determines that the input and output cannot be connected at step 410, the process returns to step 320 to allow one or more alternative components to be selected.

In this case, the user can select an alternative drafting or filing component. Alternatively, the base station 1 may include a component 53 that translates received documents from the first language to the second language, or from a Word document format to another format in which case, this component can be used to interconnect the output of the drafting component 51 and the input of the filing component 52. This is shown by the component 53 in Figure 12.

Otherwise, the user selects a connection tool and operates to generate a connection representation between the input and output of the respective component representations on the schematic representation, at step 420. The end station 3 interprets the connection representation as a connection between the respective input and output, and generates connection data representing the connection.

An example of this is shown in Figure 13. In particular, Figure 13 shows the schematic representation of the component representations 51, 52, 53 of the component representations shown in Figure 12, with the components being interconnected using the connection representations shown generally at 54.

Figure 14 shows that the component representation 51 corresponds to a compound component formed from a number of sub-components. These sub-components are in turn represented as a specification component 71, and a drawing component 72. These

components are adapted to allow the text and the drawings to be created separately. The specification component 71 includes two inputs 71A, 71B and two outputs 71C 71D. The drawing component 72 includes an input 72A, and an output 72B. This is shown for the purpose of explanation only as generally the manner in which the component 51 is implemented will be retained as secret by entity and this information would not be available to the end user.

In use, the specification component 71 receives details of the invention at the input 71A, and outputs the specification at the output 71D. Similarly, the drawing component 72 receives details of the specification and/or draft drawings at the input 72A, and outputs the drawings at the output 72B.

The function of the components represented by the schematic shown in Figure 14 is not important for the purposes of this example, and is used only to demonstrate the operation of the schematic representation.

In any event, the user can select a respective input and output on the schematic representation, and then draw on a connection representation between the inputs and outputs at step 420 thereby defining a connection between the respective input and output.

In this example, neither the end station 3 or the processing system 10 operates to examine the validity of the connections, and in particular does not determine whether data can successfully be transferred from the output of the first component, to the input of the second component. However, it will be appreciated that checking by the processing system may be performed in some implementations.

In any event, in this example, the end station stores an indication of the created

connection in the form of connection data at step 430.

The user then reviews the schematic representation and determines if further connections are required at step 440. If it is determined that further connections are required at step 450, the process returns to step 390 to allow further connections to be defined in the manner described above.

Thus effectively, the user will use the drawing tool to define all the connections required in the schematic representation. This will typically require that each input and output of each component is coupled either to another output or input. It will be appreciated that agents are not necessarily associated with a respective "output" or "input" or may be multiple transactions in both directions. The agents are therefore used to establish communications links between components.

If it is determined that no further connections are required for the components in the schematic representation at step 450, the user reviews the schematic representation and determines if more components are required at step 460. This allows the method to return to step 320 so that more components may be included, if it is determined that more components are required at step 470.

Thus, the user can effectively select two or more components and operate to interconnect these, before going back to select further components.

If it is determined that no further components are required at step 470, the user indicates that the service is to be performed at step 480.

At this point, the user may optionally review the schematic representation and determine if the service is to be performed. In particular, the user will generally be presented with

information regarding the overall service provision, such as an indication of the overall cost, time, resource usage, resultant performance, or the like.

This is typically generated by having the end station 3 and the processing system 10, cooperate to determine the relevant information. Thus for example, the end station 3 may transfer an indication of the schematic to the processing system to allow the relevant values to be determined, or the like. Alternatively, cost and time estimates are involved with the component specification, so that these values may be determined by the end station 3.

In any event, this allows the user to assess whether they are satisfied with the service defined by the respective schematic representation, and therefore whether they wish to proceed.

The implementation of the services defined in the schematic representation is achieved through the use of agents, which operate to allow the components to interact, as will now be explained in more detail.

In particular, upon receiving instructions to provide the component services at step 480, the end station 3 accesses the component and connection data at step 490. At step 500, the end station uses the component data to determine the components to be used in performing the service.

It will be appreciated that managing the construction may alternatively, or additionally be performed by the processing system 10, one or more of the entity stations 5, other processing systems, or a combination of the above depending on the respective implementation. However, the remainder of this example will be described with reference to the management process being performed by the end station 3.

At step 510 the end station 3 generates a purchase order corresponding to each component to be used. In particular, the purchase order is adapted to be sent to the entity providing the respective service, via a respective entity station 5, to request the provision of the respective component services. In general each purchase order will include at least the following information:

- Manufacturer ID
- Component ID
- Service ID – used to identify the respective component service instance as there may be more than one component on the respective schematic.
- Agent connection details - to allow the agents of the component to be connected.

It will be appreciated that whilst the above describes the use of purchase orders, these are not essential, and alternative techniques for ordering the implementation of component may be used.

At step 520 each purchase order is sent to the respective entity, allowing each entity to determine if it is capable of performing the respective component service at 530. Thus for example, an entity may become unavailable due to implementation problems such as faults with the entity station 15, or unavailability of an individual performing the service manually, or the like.

It will be appreciated that in the event that a component is formed from a number of sub-components, the inability of an entity to implement the component may arise from the failure of one or more of the sub-components, which in turn may be the responsibility of other entities.

If an entity cannot perform a service, whether this is due to a problem with the respective

component itself, or any associated sub-components, an indication of this is transferred to the end station 3. Accordingly, at 540 the end station 3 determines if all components can be performed. If not, the user updates the schematic representation by selecting one or more alternative components at step 550. Thus, for example, the process can return to step 320, to allow different components to be selected.

If an entity can perform a requested service, an indication of this will also be transferred to the end station 3, indicating a respective component instance ID, which uniquely identifies the component instance that has been assigned to perform the task. This is important as it will be appreciated that in general, each processing system 15 might be performing the same component simultaneously for a number of different service implementations. Accordingly, the use of a component instance ID ensures that data is transferred to the correct component instance for processing.

If it is determined that all the services encapsulated by all of the components can be performed, the end station 3 (or other processing system) determines respective inputs and outputs that are to be connected during the process at step 560. This is determined in accordance with the connection data.

At step 570 the end station 3 then activates each agent associated with each input and each output to be connected. In particular, the agents are activated in a predetermined sequence. As an agent is activated, the agent determines details of the respective data (or other information generally referred to as data hereinafter) formats and/or types from the respective input/output specification at step 580. At step 590 the agents then compare the determined data formats/types or perform negotiation to allow a common format to be determined.

Thus for example, the agents of the respective input and output can cooperate to compare

the respective data formats/types to determine if there are any data formats/types in common. In this regard, it will be appreciated an input and output can only successfully communicate if both the input and output are able to handle a common data format and/or type.

Thus, in the example outlined above, the agent associated with the output of the drafting component will communicate with the input of the filing component to determine if the format of the provided patent specification will be acceptable.

If it is determined that there are no data formats/types in common at step 600 the process proceeds to step 610 at which points the agents determine that the components can not be interconnected. This will occur for example if one of the components is outputting data in a first format (such as the first language) whilst the other component needs to accept the data a second format (such as the second language). If this occurs, the service implementation process is halted and the individual requesting the performance of the service is informed.

This allows the user to take corrective measures to allow the build process to continue. This may be achieved, for example by adding in additional components or agents, or my manual manipulation of the data, to allow the error to be corrected (for example to allow the specification to be translated). Alternatively, the process can be terminated such that the service is not provided.

In any event, if the respective input and output have data formats/types in common, then at step 630 the agents indicate that the respective inputs and outputs are ready to communicate. Once all the connections are ready at step 640 then the processing system 10 activates the components at step 650. In particular, the components will provide the respective component services defined therein by having the entity stations 5 interact

with data or other information. The data may be manipulated before being transferred to the end station 3, or another one of the entity stations 5, as required by the defined schematic representation.

Thus, in the above example, details of the invention will be transferred to a respective entity station 5 implementing the drafting component 51, allowing the respective entity (which will typically be a suitably qualified patent attorney) to perform the drafting of the specification. This may involve the use of a separate draftsperson in the formation of the drawings as represented by the sub specification and drawing components 71, 72.

In general, each component will be implemented at the respective entity station 5. In order to achieve this the data to be manipulated will be downloaded from the end station 3, the base station 1, or another one of the entity stations 5, to the respective entity station 5. This will be achieved by transferring the data to a specific port or the like on the processing system 15, as indicated in the component specification. The data will be transferred with the component instance ID to ensure that the correct component instance is used to manipulate the data.

It will be appreciated however that this may be achieved using other techniques, such as providing each component instance at a respective port, and transferring the data to the respective port.

In any event, when a respective component instance receives the data to be manipulated, the respective component will interact with the data, modifying the data or generating new data as required before providing the modified or new data at one or more of the output ports. Thus, in the example above, the drafting component outputs the completed patent specification at the output 51B.

The data will then typically be transferred to the base station 1 or the end station 3 for temporary storage in the memory 21 or the database 11, before being transferred to the input of the next component. Alternatively however the data provided at the output port of a component at one of the entity stations 5 could be transferred directly to another component instance implemented either on another one or the current entity station 5, for subsequent manipulation by the other component.

In any event, in the example above, the specification is transferred to the filing component 52, allowing the specification to be filed. It will be appreciated that this may be performed by the entity that performed the drafting. However, the system provides for the entity to be different. This in turn allows the drafting and filing of patents to be provided as different services by different entities, which allows users to select preferred drafting entities and preferred filing entities based on a wider range of criteria.

Thus, it will be appreciated that during this process, data may also be manipulated simultaneously by several different components depending on the format of the component specification.

Furthermore, it is usual for the base station 1, the end stations 3, and the entity stations 5 to be effectively interchangeable or implementable on a common processing system in the examples outlined above. Accordingly, the processing system 10, 15 and the end station 3 will generally execute applications software allowing the functionality of each of the base station 1, the end station 3, and the entity station 5 to be implemented. This allows an entity to use the entity station 5 to obtain a service in the manner outlined above for the end station 3, and vice versa.

Thus, for example, an entity may have a number of processing systems, some of which operate as entity stations 5, and some of which operate as end stations 3, depending on

the functionality required at the time. The entity may therefore be providing a number of component services, the implementation of which is distributed across the processing systems. In this instance the functionality provided by the processing systems will be equivalent to either or both of the end stations 3 and the entity stations 5, as required.

It will be appreciated that performing the implementation of components will typically require a support structure, and it is therefore common for the entity to have an infrastructure in place including a number of end stations 3 that will be used in supporting the implementation of the service. This may include for example the provision of patent attorneys. From this it will be appreciated that entities can often provide component services with little or no modification in operation, although the ability to specialise, and therefore be more competitive is vastly increased.

In the case of an entity providing a service, the entity would typically have a number of entity stations 5 that will be automated. However, if an exception, or other error occurs, such that the service cannot be completed, then the entity station 5 will hand-off or transfer the implementation of the service to another entity station 5 that is operated by an individual. This allows the individual to provide manual feedback to allow the exception or error to be resolved, if possible. Otherwise, an indication that the problem cannot be resolved will be returned to another component or entity within the system. Thus, if the problem occurs with a sub-component an indication of the problem will initially be returned to the parent component. This will continue with the exception being passed up the chain until it can be resolved.

Some of the features of the implementation described above, such as the nature and operation of the agents is described in more detail below.

Practical Implementation

It will be appreciated from the above that the base station 1 allows services provided by a number of different entities, typically at respective entity stations 5, to be accessed centrally by a number of different users. This allows components provided by entities to be reused a large number of times in the provision of numerous different services.

In general, the system will be implemented as a forum that provides users with access to the different services. The forum may be implemented using a single base station, as in the example described above. However, persons skilled in the art will appreciate the forum may be implemented using a number of base stations, and a number of associated processing systems, with the forum being distributed between the base stations and the user end stations 3.

The following description therefore focuses on the implementation of the system using a forum, although the techniques are equally applicable to any implementation, such as the use of a single base station.

In use, it is typical for each entity to define a fee associated with each component. This fee corresponds to a fee payable by users of the forum, for the use of a respective component instance service. Thus, the users pay one or more fees to each entity in return for the provision of one or more services provided by the entity.

This allows the entities to charge a fee for the provision of the respective services, thereby allowing the entities to obtain income to recoup the investment made in the development of the respective components. This in turn allows entities to specialise by providing, and obtaining financial return for, specific well-defined services.

This in turn allows entities to focus on optimisation of specific service provision, rather than trying to provide a wide range of service.

By having the forum provide users with access to a number of components, provided by different entities, and which provide similar services, this will force entities to compete against each other to provide similar services to the user. The resulting market forces will therefore drive competition between the entities, thereby forcing each entity to improve the provision of its respective service in order to capture greater market shore.

In particular, users will tend to select components that are deemed to be more successful. As a result, entities compete with each other at the component level to provide more and more successful components. This allows the entities to invest more time and money in improving the implementation of the specific components, whilst recouping the investment as more successful components will be purchased a larger number of times.

In this regard, components may be deemed to be more successful if they are cheaper, faster, result in more optimal code, or the like, when compared to other components offering the same service.

From this, it can be seen that market forces and direct competition at a specialisation level will lead to improvement in each service provided through the forum. Thus, each component at every level within the hierarchical structure will be optimised resulting in the performance of services in a more efficient manner. This is enhanced by the standardisation in interaction between different entities through the use of the component specification. In particular, the user selects components that are capable of interacting based on the component specifications, during the schematic creation stage, thereby allowing majority of communication to be performed automatically.

In addition to this however, users can select components in accordance with other criteria. In order to help competition within the forum, the user will be provided with information to allow an assessment of the best components for use in constructing the respective application. The user can then select components in accordance with a wide variety of factors including, for example:

- The entity performing the respective service;
- The cost;
- The location of the entity performing the respective service;
- The popularity of the component;
- The data format/types that can be received by or output from the component; and,
- Ratings given to the component by previous users of the forum or the forum operator.

In this regard, the forum will generally provide a rating system allowing users to rate the effectiveness of components. Ratings can be determined statistically, for example by determining the number of build faults that occurred for each respective component, by user feedback, or by testing of the components by the forum itself.

It will be appreciated that whilst market competition through the use of reviews or the like exist, this is normally provided for entire entities, even though the entity may provide many different services. In contrast, the review and rating in this instance is performed at the component level thereby forcing the improvement of individual component services, as opposed to the entity as a whole.

It will be appreciated that other factors may also be used in judging the success of components.

In any event, in order to remain competitive, each entity will focus on providing well-

defined, efficient service implementations, thereby allowing the services to be provided in a more efficient manner.

In order to allow the operators of the forum to make a profit, it will also be typical for at least a portion of any fees charged by the entities, to be provided to the operator of the forum, allowing the operator to obtain profit based on usage levels of respective components. However, alternatively, subscription charges or the like could be applied to individuals wishing to use the system, or registration fee for entities wishing to submit components to the forum.

Further details of the example of the implementation of the processes outlined above will now be described in more detail below.

Agents

In this example, agents are the only form of inter-component communication. Agents are responsible for providing and gathering all the information a component needs to complete the service it embodies. An agent is often formed from a simple piece of executable code with limited functionality, which may also be supported with manual interaction as required.

In particular, when the forum, and in particular the end station 3 sends out purchase orders to the entity stations 5, the purchase orders contain agent connections specifying how the agents of the respective components should interconnect.

The reception of a purchase order causes the entity station 5 to implement a new component instance, and corresponding agents that are capable of finding and connecting to the agents of other components. The agents only ever connect to (communicate with) other agents.

Whilst the agents are often only simple, processes called bundling and hand-off allow agents to exhibit complex behaviour and powerful information providing and gathering capabilities.

The hand-off mechanism terminates an agent to agent transaction and opens another. This is most useful when a component is using the agent of a sub-component as if it were an agent on the component itself. Worked examples help to clarify the hand-off procedure and an example of hand-off to a sub-component is presented.

Bundling is a recursive mechanism by which multiple agents related by a specific purpose can be treated as a single simple agent. Worked examples help to clarify the bundling and debundling mechanism and an example of bundling and debundling components are presented.

In particular, an example will now be described with reference to Figure 15, which shows a schematic P having components X and Y connected by agents X1 and Y1.

In order to specify the address of a particular agent for a particular component instance, it is necessary to be able to identify the agent uniquely. Accordingly, for the purposes of this example, the component X is manufactured by an entity having a manufacturer ID IDx, and component Y is manufactured by a an entity having a manufacturer ID IDy.

When schematic P is laid out, the symbols for components X and Y are downloaded, arranged and connected in the schematic representation P using the method described above with respect to Figures 9A to 9E. As the schematic is constructed, each component is given a unique label, as shown at U1 and U2. These labels allow

schematics with more than one component of the same type to reference the correct instance of that component.

Associated with the respective component representations are the component specifications, including the manufacturer ID, and part number. This allows the user end stations to generate and send out the purchase orders to the corresponding entities. The component symbols are also associated with input and output specifications detailing the agents, which for this example is exactly one for each component.

Before the schematic can be set out for construction, the user creating the schematic P must be specified so that the entities IDx, IDy can bill the user. In this example, the user is given an identifier IDp. Once this information is contained in the schematic it is ready to be built.

The process of building a schematic results in a number of entities being contracted with purchase orders. Thus the decision to build will incur costs and contractual responsibility. A mistake in the schematic may result in a bad build wasting time and money.

Assuming the build is to proceed, the next step is to submit schematic P to the builder program which will typically be implemented by the end station 3. The builder program interprets the schematic and compiles and issues purchase orders for each component in the schematic. The purchase orders for the component X would contain the following information:

- Base station identifier 1
- Schematic identifier P
- X component label U1
- Entity identifier IDx

- Entity part number X
- Component X agent connection details, including:
  - Entity identifier IDy
  - Entity part number Y
  - Component Y agent number
  - User's identifier IDp
  - Schematic identifier P
  - Y component label U2

The purchase order for the component Y would include similar information.

Should the component being purchased have more than one agent, then each agent must have separate connection details included with the purchase order specifying the agent's target.

When the entity IDx receives the purchase order from the user IDp, the entity IDx creates an instance of its component to satisfy the requirements of the purchase order. This is achieved using a constructor, the operation of which will be described in more detail below.

After the provision of the purchase orders, the respective instance of the component X now has the information it needs for the agent X1 to communicate with the agent Y1. Accordingly, the agent X1 connects to the entity station 5 of the entity IDy and requests the agent Y1, using the information received in the respective purchase order, as outlined above.

The agents X1 and Y1 connect and then proceed to authenticate using the information already known about each other.

After the connection and authentication is complete, the agents X1, Y1 can negotiate to determine the information needed for each to complete construction of their respective components.

Thus, for example, in the event that the agents are adapted to handle the data types shown in Figure 16, the agent X1 can handle integers INT, characters CHAR, and floating point numbers FLOAT, whereas the agent Y1 can handle double inputs DOUBLE, floating point numbers FLOAT, and dates DATE. Accordingly, the agents will determine the component X must provide the output in the form of floating point numbers FLOAT.

From the above it will be appreciated that in this example the agents are the only form of inter-component communication. As a result the agents must be able to communicate via the communications networks 2, 4.

Thus in the example described above, the agent of the output 51B will operate to connect to the agent of the input 52A. The agents will determine how the patent specification is to be transferred, and this may therefore represent no more than the provision of an e-mail address or the like, to allow the specification to be e-mailed from the entity providing the drafting services to the entity providing the filing services. Alternatively more complex information may be transferred, such as the desired format of the specification, any security passwords to be used to protect the document, or the like.

A number of features of the implementation of the agents will now be described.

Hand-Off

In addition to the agents making a static connection there exists an agent hand-off mechanism. The agent hand-off mechanism allows an agent to agent connection to terminate with one of the agents reconnecting to yet another agent.

This behaviour provides a means by which agents can exhibit seemingly complex behaviour with simple agents. This is achieved by having an agent perform some simple negotiation then hand off to another agent to negotiate further. With only these simple transactions a complex overall transaction can occur with the seemingly simple agents.

In particular, handoff allows a component to present a service that internally is made up of carefully crafted arrangement of sub-components. Thus although a component presents a number of agents and appears to perform a given service, in actual fact the service is supplied by many sub-components which have at least some of their agents satisfied by a handoff from a parent agent. In order to achieve this an agent of the parent must first connect to some outside component, then order that agent at the other end to connect to a sub-component.

A walk through of the handoff mechanism demonstrates the steps involved in the handoff process.

Figure 17A represents a schematic involving a handoff. In Figure 17A agent A1 of component C1 and agent A2 of component C2 connect as normal, however C2 has sub-component C3 as shown in Figure 17B.

The component C2 intends to handoff the agent A1 to the agent A3 of the sub-component C3 as shown in Figure 17C.

However to simplify the agent protocol, the agent can connect to more than one agent at a time. Thus the agent A2 could not connect to the agents A1 and A3 at the same time. To allow the component C3 to be built and access to the agent A3 to be gained, a temporary agent A2b is created as shown in Figure 17D.

This has the benefit of allowing the component C3 to be built and the agent A3 to connect to an agent providing means for the component C2 to communicate to the component C3. When the agent A1 is connected to the agent A2 and the agent A2b is connected to the agent A3, the component C2 can direct the agents A2 and A2b to terminate and cause the agent A1 to reconnect to the agent A3

Thus, the component C2 uses the local agents A2 and A2b to communicate the hand-off order to the agents A1 and A3 respectively, resulting in the agents A1 and A3 connecting as shown in Figure 17E.

An example of the handoff mechanism handing from one component to another will now be described with reference to Figure 18. In particular, in this example, the schematic includes three components A, B, C, each of which has respective agents A1; B1, B2; C1, C2.

In this example, the agent A1 gets the address of the agent B1 from the schematic purchase order. Similarly the agent B1 gets the address of the agent A1 from its schematic purchase order. Using the agent connection mechanism outlined above agents A1 and B1 connect and authenticate.

Agents A1 and B1 perform their information transfer, and this results in the agent B1 deciding to hand off the agent A1 onto the agent C1. The component B obtains the agent address of the agent C1 by having the agent B2 communicate with the agent C2. The

agent B1 then uses its authenticated link to the component A, and sends a hand-off request together with the agent address of the agent C1.

The agent A1 simply disconnects from the agent B1 and connects with the agent C1. Should the agent C1 be busy with a connection elsewhere. The agent A1 simply waits for the agent C1 to become available. Similarly the component B obtains the address of the agent A1 and transfers this to the agent C1, allowing the agent C1 to reconnect to the agent A1.

Bundling

Often a component will require a number of agents to resolve information for a specific task. Since these agents are related it makes sense to group the agents into a bundle to hide the complexity and deal with the bundle like a single agent. This greatly simplifies the schematic and reduces errors.

Thus, the purpose of the bundling is to manage agents more effectively. Although not strictly necessary bundling allows related agents to be attached to each other so that its relationship is preserved making the management of large numbers of agents an easier task.

Special components provide the service of bundlers/debundling and these will hereinafter be referred to generally as Bundlers. In this example, bundlers have three agents - two "inputs" and an "output", whereas bundlers operating to debundle (which may be referred to as "debundlers") have two "outputs" and an "input". The terms output and input are inverted commas as the bundler and debundler perform almost exactly the same task. They both gather the addresses of a pair of agents and send it through a third agent, however the bundler is the one that initiates the communication, and so gathers the addresses first - making the pair of agents inputs and the lone agent an output. As soon

as the debundler has received the addresses through its input the roles are reversed. Once the bundler and debundler have swapped the information, they handoff the connected components to each other. If either of these connected components is a bundler or debundler, the process begins again.

Figure 19A represents two agents A1, B1 from respective components A, B coupled through a bundle component BUNDLE, which provides bundle agents BU1, BU2, BU3. The bundle agent BU3 is used to connect to the component X. The bundle agents BU1, BU2, BU3 are indistinguishable from a normal agent.

In use, the component BUNDLE depicted in Figure 19A receives connections from the agents A1, B1 and presents the agent BU3. The role of the agent BU3 is to provide the addresses of the agents A1, B1 to the component X.

In the example shown in Figure 19A, the components A, B, X receive addresses of the agents BU1, BU2 and BU3 respectively from the schematic purchase order. Similarly the bundle component BUNDLE gets the agent addresses A1, B1 and X1 from a respective schematic purchase order. The agents A1, BU1 connect and authenticate while the agents B1, BU2; and, X1, BU3 do the same. The component X negotiates with the component BUNDLE and determines that the payload of the agent BU3 represents a bundle.

Accordingly, by using the hand-off mechanism as described above, the component X can determine the addresses of the agents A1, B1, and order the bundle component BUNDLE to hand-off A1, B1 as shown in Figure 19B. Thus, in Figure 19B the component X through agent BU3 learns of the agents A1, B1. The component X then orders the bundle component BUNDLE via the agent BU3 to hand-off the agents A1, B1 to the agent X1

and X2 respectively. The bundle component and its respective agents has then completed its service and can retire.

The bundling component BUNDLE as shown in Figure 20 has no concern as to the nature of the agents A1, B1. As far as the bundling component is concerned, they are any two agents and their payload is irrelevant. This enables cascading of bundling components as shown in Figure 20. Any number of bundling components may be cascaded.

In addition to providing bundle components for bundling agents, debundling components are provided for performing the opposite function.

An example of this is shown in Figure 21, in which a component A is coupled to a debundling component DEBUNDLE, which in turn is connected to two components X, Y, as shown. In this example, the debundling process starts with the component agents A1, DBU1; DBU3, X1; and DBU2, Y1 connecting and authenticating as specified in the purchase order.

The debundling component then learns the addresses of the two agents represented by the bundle. It then requests X1 and Y1 to hand off in accordance with the addresses provided by agent A1. At this point the debundling agent has then completed its service and can retire.

An example of the manner in which bundles of agents can be used will now be described with reference to the example shown in Figure 22A.

Internal to a Component P, a Bundler 1 is connected to a Component A and a Component B. The output of the Bundler 1 is connected to one of the inputs of a Bundler 2. The

other input is connected to a Component C. The output of the Bundler 2 is the bundle of agents that will be connected to an external agent of the Component P.

The Component P is connected to another Component Q, and the bundled agent is, internal to the partner component, attached to a Debundler 1. The Debundler 1 is attached to a Component D and a Debundler 2. The Debundler 2 is attached to a Component E and a Component F.

During automatic handoff, the Bundler 2 will be connected to the Debundler 1. The bundler 2 obtains the address of the agent connected to the Component C and the Bundler 1 respectively and sends these to the Debundler 1, which will respond with the address of the agent connected to the Component D and the Debundler 2 respectively. The Bundler 2 and the Debundler 1 will then perform a handoff of this respectively agents so that the Component C will be connected to the Component D, and the Bundler 1 will be connected to the Debundler 2 as shown in Figure 22B. The Bundler 2 and Debundler 1 are now finished, and retire.

Now the Bundler 1 will send the addresses of the agents connected to the Component A and the Component B to the Debundler 2, which will respond with the addresses of the agents connected to the Component E and the Component F. The Bundler 1 and the Debundler 2 will then perform a handoff so that the Component B will be connected to the Component E and the Component A will be connected to the Component F. The Bundler 1 and the Debundler 2 are now finished, and retire as shown in Figure 22C.

Some very complex patterns of bundling and debundling can be used provided that the patterns are symmetrical around the link going between the bundlers and the debundlers. These patterns can span across multiple components and multiple levels of the component hierarchy symbolised in Figure 22D.

Bundle Arithmetic

In this example, there are just two primitive components used in agent arithmetic. They are the:

- Bundle Component.
- Debundle Component.

In this section two examples of bundle arithmetic are given. These are:

- Bundle Reordering
- Rake-out

With components bundle and debundle a number of useful operations can be performed on bundles. An example of agent reordering can be found in Figure 23. In this example, the component U represents a debundle (unbundle) component, with the component B representing a bundle component. Accordingly, in this example, the agents a, b are reordered within the bundle by extracting the agents a, b from the bundle and reassembling the bundle with the agents b, a in the reverse order.

Rake-out is when an agent is extracted from a particular agent bundle. An example of agent rake-out can be found in Figure 24. In this example, a respective agent c in a bundle of agents a, b, c, d is required to be separated from the bundle. This is achieved by debundling the bundle using the debundle components U until the agent of interest is available, then rebundling the bundle using the bundling components B.

**Specific Examples**

A more detailed specific example will now be described with reference to Figure 25.

In particular, Figure 25 shows further development of the schematic representation of

Figure 10 to allow foreign patent applications to be filed. In this example the schematic representation includes fourteen basic components 81, 82, 83, ... 94, each of which includes respective inputs and outputs as shown.

In this example, the component service provided by each component is outlined in Table 1 below, with the function of the inputs and outputs being indicated in Table 2.

Table 1

| Component | Name | Component Service |
|---|---|---|
| 81 | Drafting | Preparing patent Specification |
| 82 | Provisional Filing | Filing a Provisional Application |
| 83 | US Application | Obtaining a US patent |
| 84 | European Application | Obtaining grant of a European Application |
| 85 | UK Validation | Validating the European Application in the UK |
| 86 | French Translation | Translating documentation into French |
| 87 | French Validation | Validating the European Application in France |
| 88 | German Translation | Translating documentation into German |
| 89 | German Validation | Validating the European Application in Germany |
| 90 | Japanese Translation | Translating documentation into Japanese |
| 91 | Japanese Validation | Obtaining a Japanese patent |
| 92 | Granted | Registering details of Granted Patents |
| 93 | Renewals | Policing renewal payments |
| 94 | Monitoring | Monitoring for infringing products |

Table 2

| Input/Output | Function | Input/Output Nature |
|---|---|---|
| 80A | Input | Invention Details |

| | | |
|---|---|---|
| 80B | Output | Granted Patent Details |
| 80C | Output | Renewal Reminders |
| 80D | Output | Infringement Details |
| 81A | Input | Invention Details |
| 81B | Output | Patent Specification |
| 82A | Input | Patent Specification |
| 82B | Output | Provisional Application Details |
| 82C | Output | Provisional Application Details |
| 82D | Output | Provisional Application Details |
| 83A | Input | Provisional Application Details |
| 83B | Output | US Patent Details |
| 84A | Input | Provisional Application Details |
| 84B | Output | European Patent Details |
| 84C | Output | European Patent Details |
| 84D | Output | European Patent Details |
| 85A | Input | European Patent Details |
| 85B | Output | UK Patent Details |
| 86A | Input | European Patent Details |
| 86B | Output | French Translation of European Patent Details |
| 87A | Input | French Translation of European Patent Details |
| 87B | Output | French Patent Details |
| 88A | Input | European Patent Details |
| 88B | Output | German Translation of European Patent Details |
| 89A | Input | German Translation of European Patent Details |
| 89B | Output | German Patent Details |
| 90A | Input | Provisional Application Details |
| 90B | Output | Japanese Translation of Provisional Application Details |

| | | |
|---|---|---|
| 91A | Input | Japanese Translation of Provisional Application Details |
| 91B | Output | Japanese Patent Details |
| 92A | Input | US/German/French/UK/Japanese Patent Details |
| 92B | Output | Patent Details |
| 93A | Input | Patent Details |
| 93B | Output | Renewal Information |
| 94A | Input | Patent Details |
| 94B | Output | Infringement Details |

Accordingly, in this example, the user of the end station 3 desires patents to be obtained in US, Germany, France, UK and Japan. The user selects the components outlined above and arranges these to form the component schematic shown in Figure 25 using the techniques outlined above.

It will be appreciated that this will require the user to have an understanding of the patent system. However, this may be overcome by providing a single component 80, which the user can select to obtain the required patents, with the components 81, ... 94, representing sub-components.

In any event, once the schematic has been constructed, the user can instruct the schematic to be implemented in the manner described above. Accordingly, the base station 1 or the end station 3 will generate purchase orders which are transferred to respective entity stations 5 to activate component instances thereon, and begin the agent negotiation process.

Once this is completed, the service can be performed.

Thus, for example, the agent for the input 80A will communicate with the agent 81A to determine the form of invention details that are required by the drafting component 81 to prepare the specification. This will typically depend on a number of factors such as the subject matter, the complexity of the invention or the like. Furthermore, the agents will negotiate to where the invention details should be transferred, which will also typically depend on the above mentioned factors.

Accordingly, in this instance, the agent 81A will typically be adapted to supply a list of questions regarding the invention to the agent 80A. The user will be prompted by the agent 80A to provide appropriate responses, which are then transferred to the agent 81A for analysis. Following this, the agent 81A will transfer a list of requirements to the agent 80A, together with contact information, allowing the invention details to be submitted to the entity implementing the component 81. In this case, for example, this will typically be a patent attorney, and the agent 81A will therefore provide an e-mail address or the like of a selected attorney. Alternatively, the components can agree to transfer using a particular protocol and hand-off these objects to respective sub-components skilled in that protocol, as will be appreciated by persons skilled in the art.

It will be appreciated that in order to implement this, the component 81 may include a number of sub-components corresponding to different attorneys, with the respective sub-component being selected by the agent 81A, in accordance with responses and/or invention details received from the user, via the agent 80A.

Having received the invention details, the entity will prepare the patent specification as required.

Once this is completed, or simultaneously with the above mentioned procedure, the agent 81B will communicate with the agent 82B to establish a connection between the

components 81, 82. In this instance, the filing component 82 may be the same entity as above (in which case the agent communication will establish this and end the process), or a different entity (in which case the agents 81B, 82A will establish a communication channel such as via e-mail). When the patent specification is completed, this is supplied by the component 81 to the component 82 for subsequent filing as a provisional application.

In this case, it will be appreciated that the attorney drafting the specification can save the completed specification in a predetermined location on the respective entity station 5, allowing the drafting component instance to automatically transfer the completed specification to the filing component 82. This allows the attorney performing the drafting process to focus solely on the drafting, avoiding the need to check that the specification is correctly filed as this is, in this example, no longer their responsibility.

Similarly, the filing component 82 will receive the patent specification, allowing the responsible entity to file the provisional application. It will be appreciated that this may be an automated system and need not require manual intervention, depending on the circumstances.

Once the provisional application has been filed, details of this, including a copy of the patent specification may be passed onto the components 83, 84, and 90. In this example, the filing component 81 provides a single output for providing both details of the provisional application and the filed specification to the subsequent components, whereas in the example described above, the component 52 includes two outputs, one for the details and the other for the specification. It will therefore be appreciated that these represent two alternative filing components which operate in a slightly different manner, as reflected by the difference in the component structure.

The components 83, 84, are responsible for filing and prosecuting patent applications in the US and Europe respectively. These components would therefore typically be implemented by patent attorneys within the respective jurisdictions. It will be appreciated however, that again, each of the components 83, 84 may contain several sub-components, with various aspects of the filing and prosecution being sub-contracted to different entities, or different individuals or automated processes within the entity.

This automated distribution of the work in accordance with the defined schematic vastly reduces the amount of work required by the entities and any individuals within the entities, allowing the entities and individuals to focus on specific well defined tasks, rather than a wide range of tasks. This, in turn allows the entities and individuals to focus on providing improved service in those well defined task areas.

In the example above, the translation component 90 operates to translate the received provisional application details and specification, before transferring the translated details to the component 91 for subsequent filing and prosecution as a Japanese patent application. It will be appreciated that as translation software becomes more effective, the translation components could be implemented automatically depending on legal requirements.

In the case of the European application, once this is granted, it then needs to be validated in countries of interest, which in this case includes UK, Germany and France. Accordingly, this will typically be performed by entities in the respective countries, with translations being obtained as required.

In this example, once the patents have been granted, details are transferred to the granted component 92, which distributes details of the granted patents (typically including a copy of the granted specification) to the user via the output 80B, and to the renewal and

infringement components 93, 94. The renewal component 93 calculates the renewal fees to be paid and alerts the user to these via the output 80C. Similarly the infringement component 94 operates to monitor competing products and the patent to determine any potential infringements, which can then be notified to the user.

However, the schematic may be implemented using one-to-one connections. In order to achieve this, the outputs of the components 87,85,89 could be combined by a bundler component, to allow a single bundle to be presented to the component 92. Similarly, the component 92 may be coupled to a duplicate component, adapted to duplicate the output provided at the output 92B and provide this to the components 92, 94, again via one-to-one connections.

It will be appreciated that the above described example is a major simplification on the schematic that would normally be used, not least because user feedback would normally be required at a number of stages throughout the process. However, it will be appreciated that this can be achieved through the use of suitable components, one of which is shown additionally at 95. In this case, the component 95 is a feedback component used to provide feedback to the components 83, 84, 91, during prosecution. This allows the user to have control over the prosecution of the application, for example by deciding whether to proceed with the case in the light of relevant prior art, to suggest claim amendments, or the like.

Accordingly, in this case, the connections between the components 95, 83, 84, 91 are two way connections allowing two way communication.

It will be appreciated that additional features may also be provided, such as making the implementation of certain components dependent on the completion of previous components in the schematic. Thus for example, if the patent applications are not granted

due to the presence of prior art, then there is no need to implement the components 93, 94. This can be used to reduce the amount of work performed by the system, with the components 93, 94 only being implemented when required.

Furthermore, the components may be adapted to retain control over the process for a predetermined amount of time. Thus, for example, the filing component may retain the filing details and specification for ten months, such that the European, US and Japanese applications are only filed near the end of the priority year.

A further feature shown by the schematic in Figure 25 is that some inputs and outputs are one-to-one in the sense that each input connects to one output. In contrast however, other inputs or outputs, such as the input 92A and output 92B are many-to-one, or one-to-many respectively, in that each input or output may connect to many outputs or inputs respectively. Again, this will depend on the respective implementation of the components.

It will be appreciated that similar schematics may be constructed for an almost limitless range of service industries. This includes industries such as law, accountancy, financial services, insurance, management consultancy, or the like. This list is not intended to be exhaustive, and the use of suitable components will allow the techniques to apply to any industry.

Persons skilled in the art will appreciate that numerous variations and modifications will become apparent. All such variations and modifications that become apparent to persons skilled in the art, should be considered to fall within the spirit and scope that the invention broadly appearing before described.

Dated this twenty second day of April, 2003

**NOEL WILLIAM LOVISA**

By his Patent Attorneys

**DAVIES COLLISON CAVE**

5

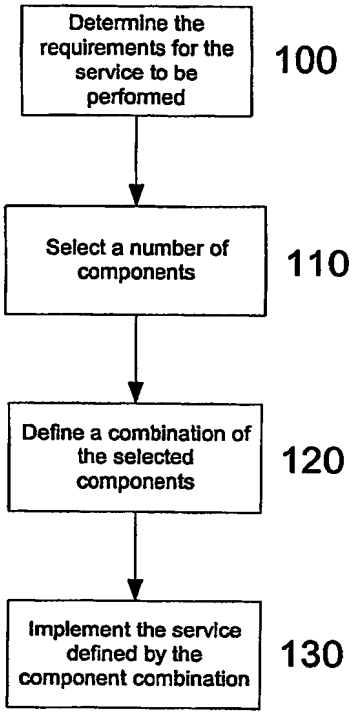
Determine the requirements for the service to be performed
100
Select a number of components
110
Define a combination of the selected components
120
Implement the service defined by the component combination
130

Fig. 1

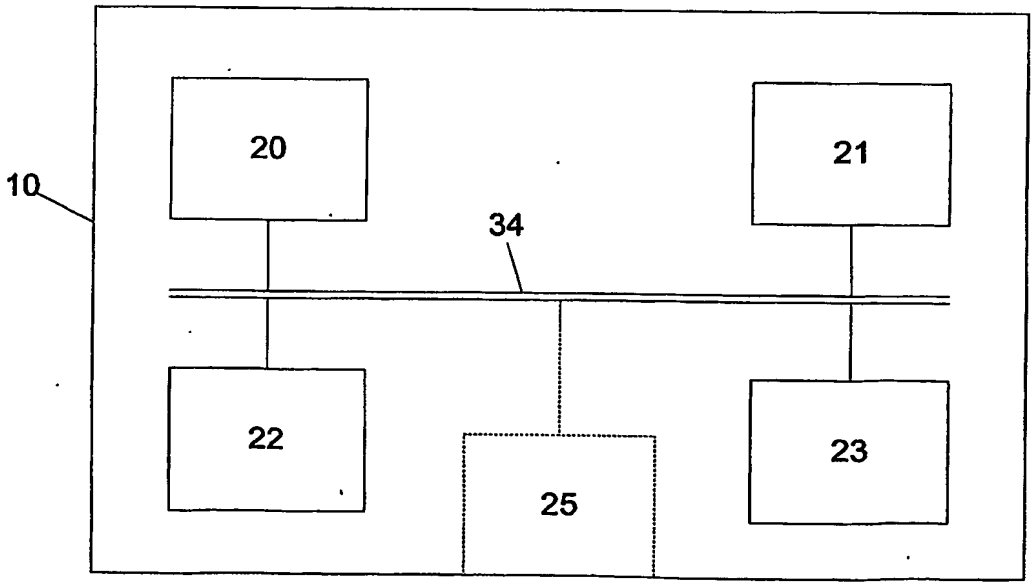
10
20
21
34
22
25
23

Fig. 3

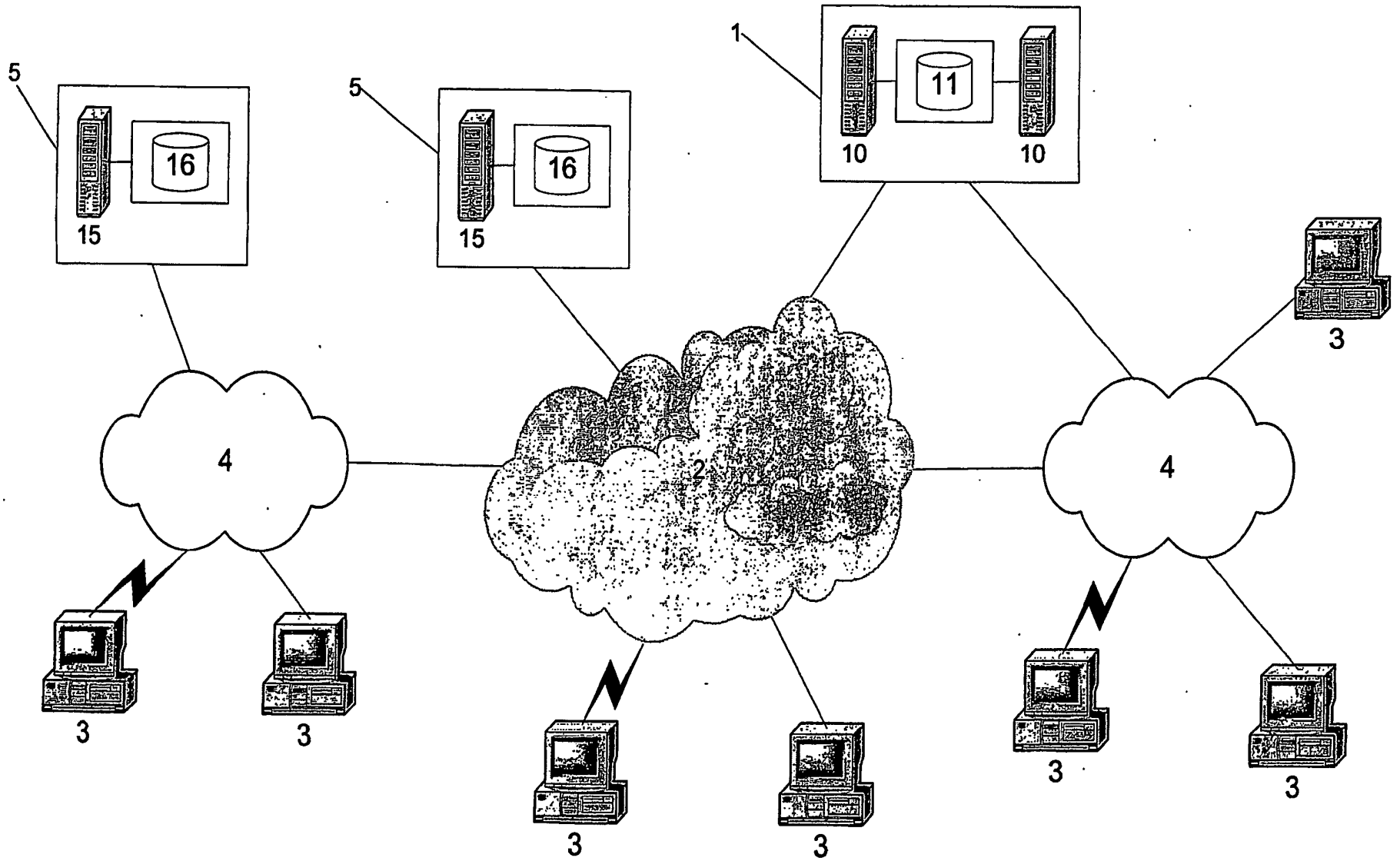


Fig. 2

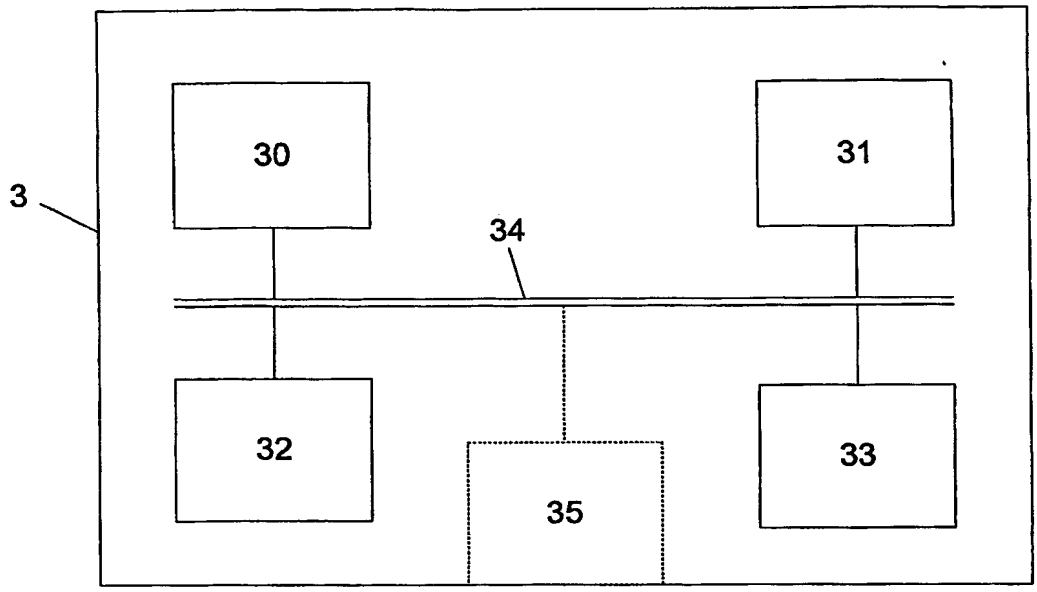


Fig. 4

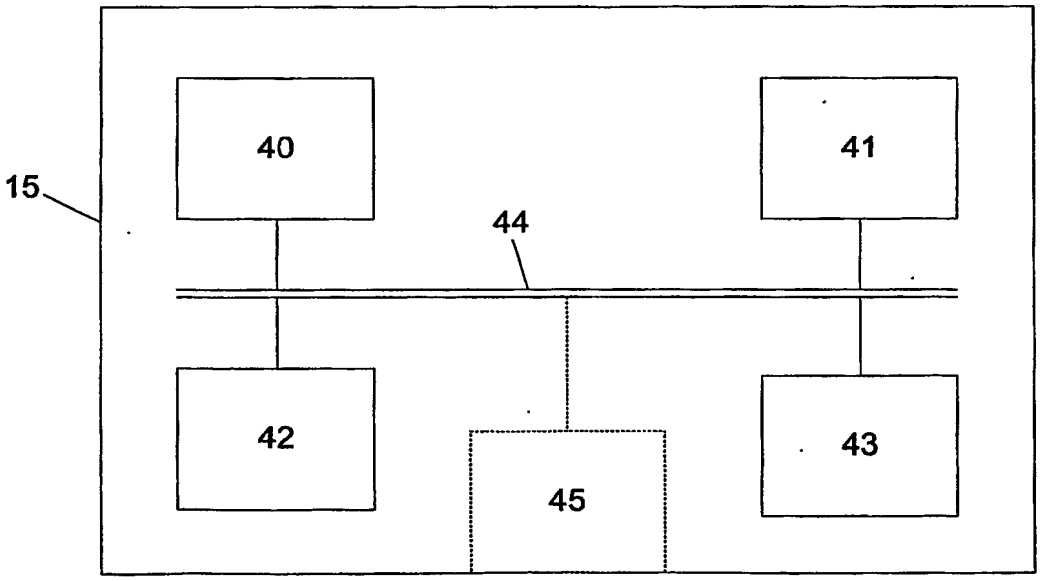


Fig. 5

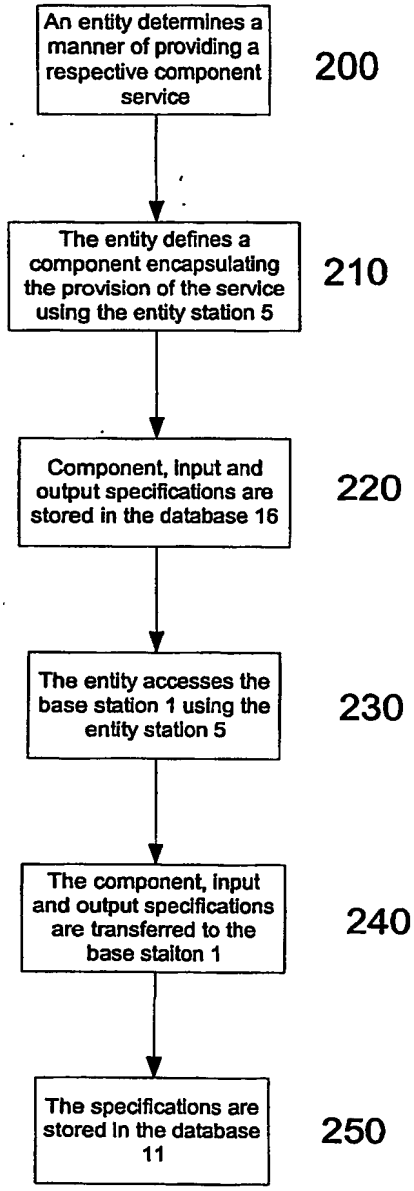
An entity determines a manner of providing a respective component service
200
The entity defines a component encapsulating the provision of the service using the entity station 5
210
Component, input and output specifications are stored in the database 16
220
The entity accesses the base station 1 using the entity station 5
230
The component, input and output specifications are transferred to the base staiton 1
240
The specifications are stored in the database 11
250

Fig. 6

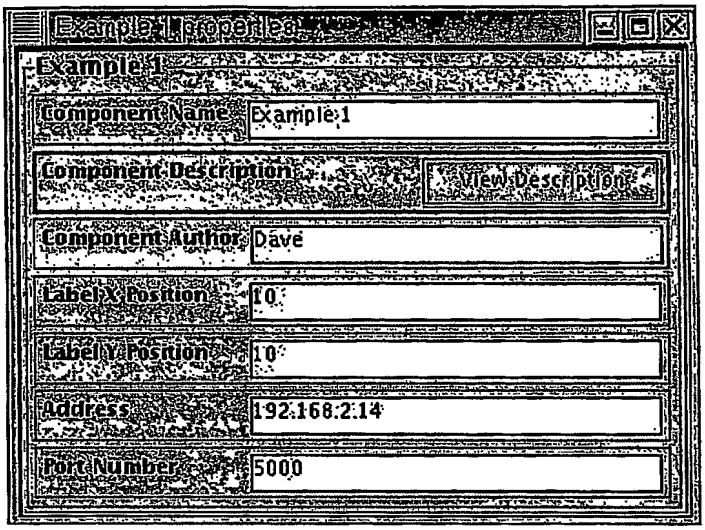
Example 1 properties
Example 1
Component Name
Example 1
Component Description
View Description
Component Author
Dave
Label X Position
10
Label Y Position
10
Address
192.168.2.14
Port Number
5000

Fig. 7

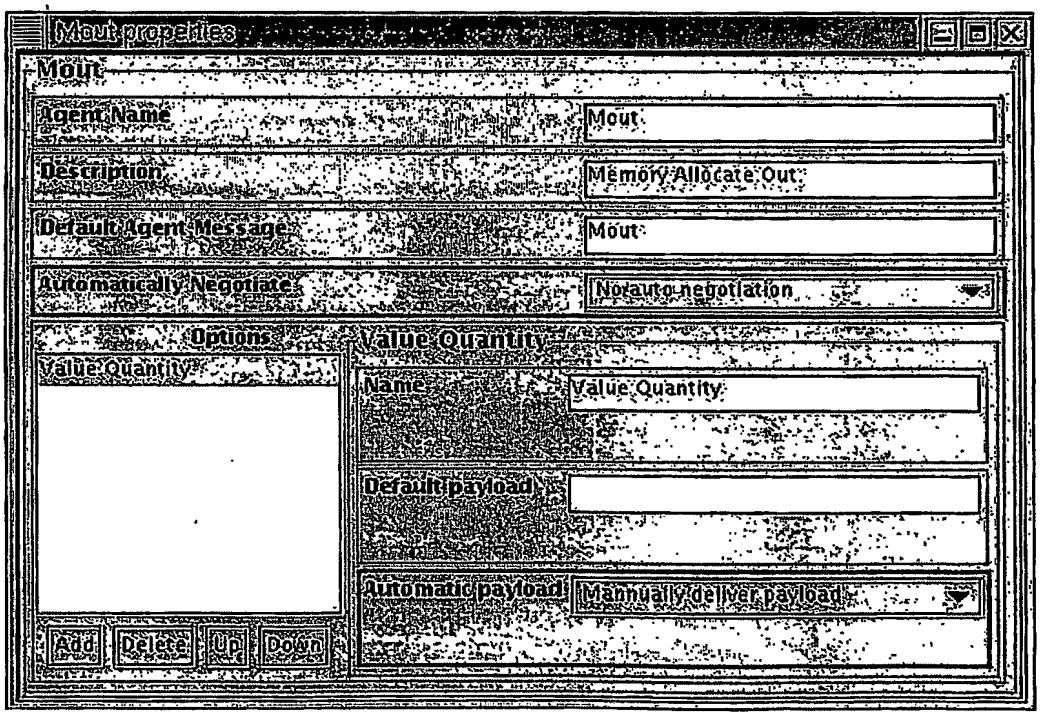
Mout properties
Mout
Agent Name
Mout
Description
Memory Allocate Out
Default Agent Message
Mout
Automatically Negotiate
No auto-negotiation
Options
Value Quantity
Value Quantity
Name
Value Quantity
Default payload
Automatic payload
Manually deliver payload
Add
Delete
Up
Down

Fig. 8

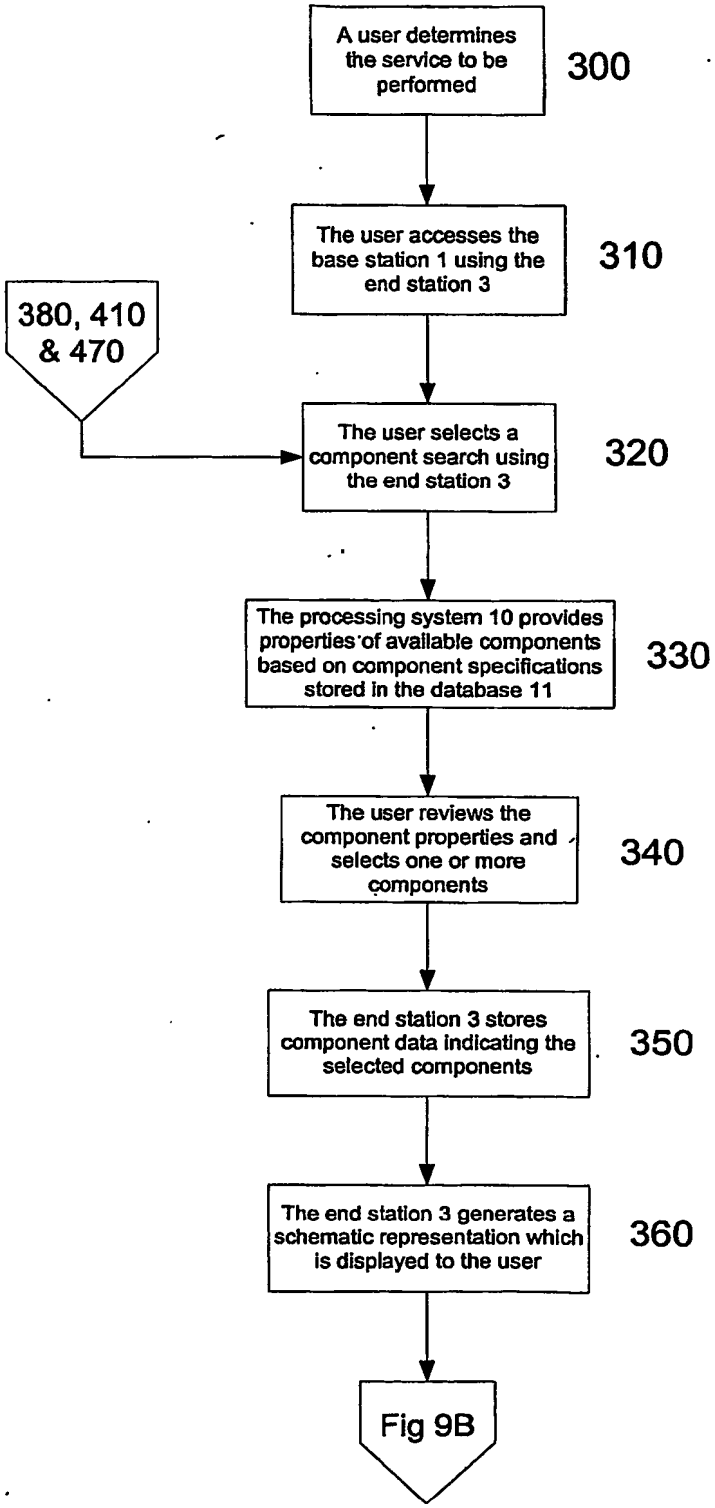
A user determines the service to be performed
300
The user accesses the base station 1 using the end station 3
310
380, 410 & 470
The user selects a component search using the end station 3
320
The processing system 10 provides properties of available components based on component specifications stored in the database 11
330
The user reviews the component properties and selects one or more components
340
The end station 3 stores component data indicating the selected components
350
The end station 3 generates a schematic representation which is displayed to the user
360
Fig 9B

Fig. 9A

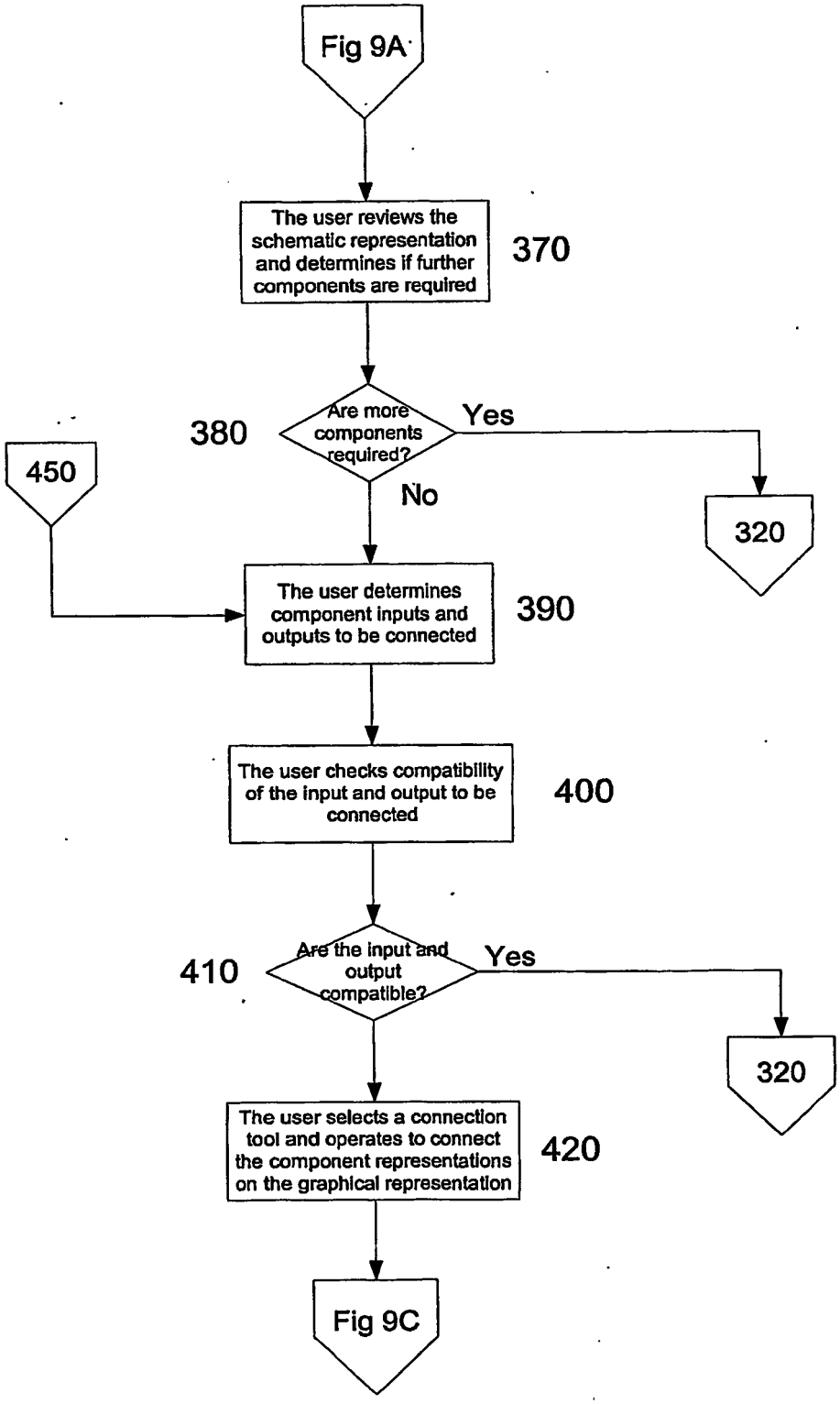
Fig 9A
The user reviews the schematic representation and determines if further components are required
370
380
Are more components required?
Yes
No
450
320
The user determines component inputs and outputs to be connected
390
The user checks compatibility of the input and output to be connected
400
410
Are the input and output compatible?
Yes
320
The user selects a connection tool and operates to connect the component representations on the graphical representation
420
Fig 9C

**Fig. 9B**

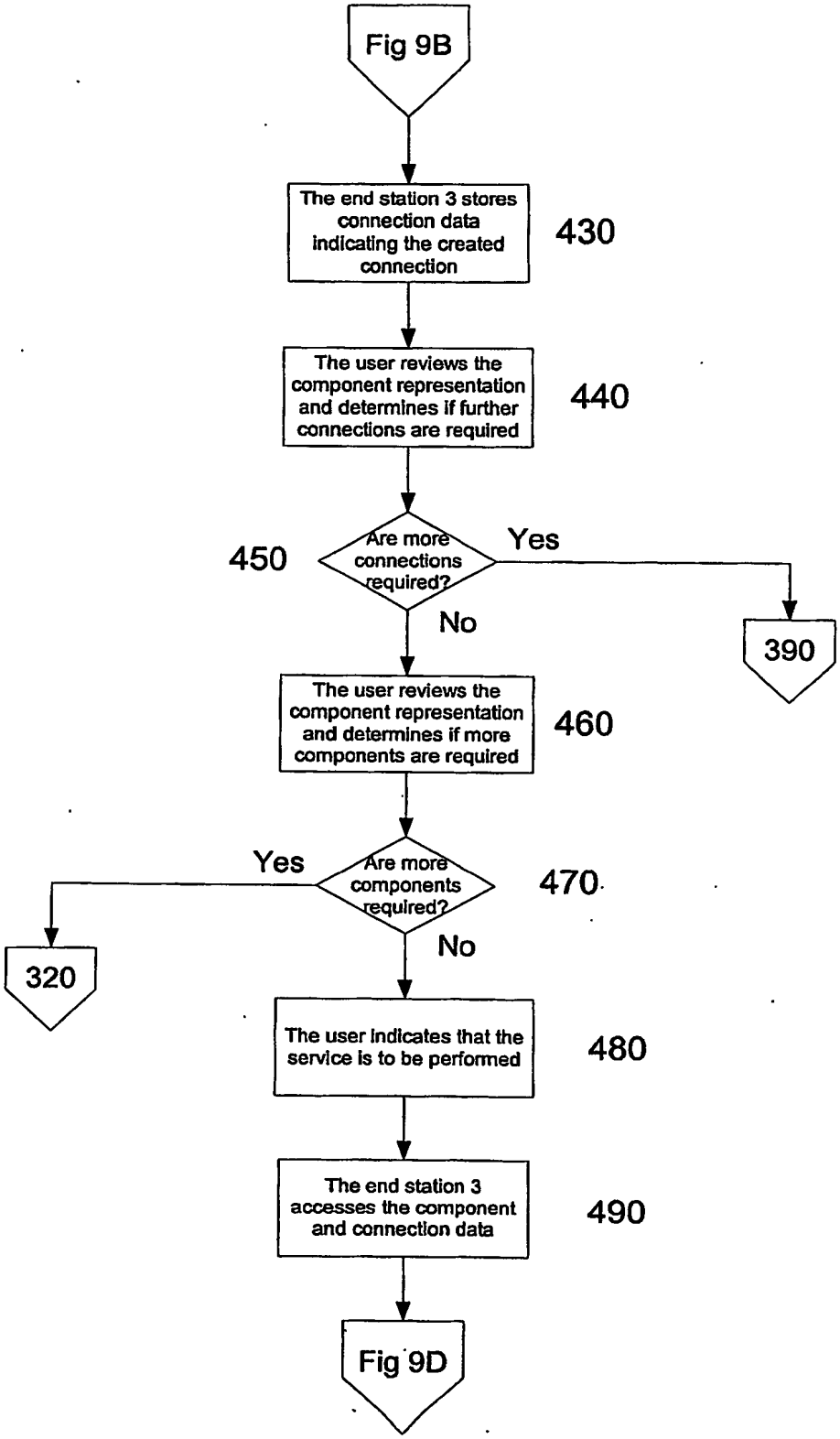
Fig 9B
The end station 3 stores connection data indicating the created connection
430
The user reviews the component representation and determines if further connections are required
440
450
Are more connections required?
Yes
No
390
The user reviews the component representation and determines if more components are required
460
Yes
Are more components required?
470
No
320
The user indicates that the service is to be performed
480
The end station 3 accesses the component and connection data
490
Fig 9D

Fig. 9C

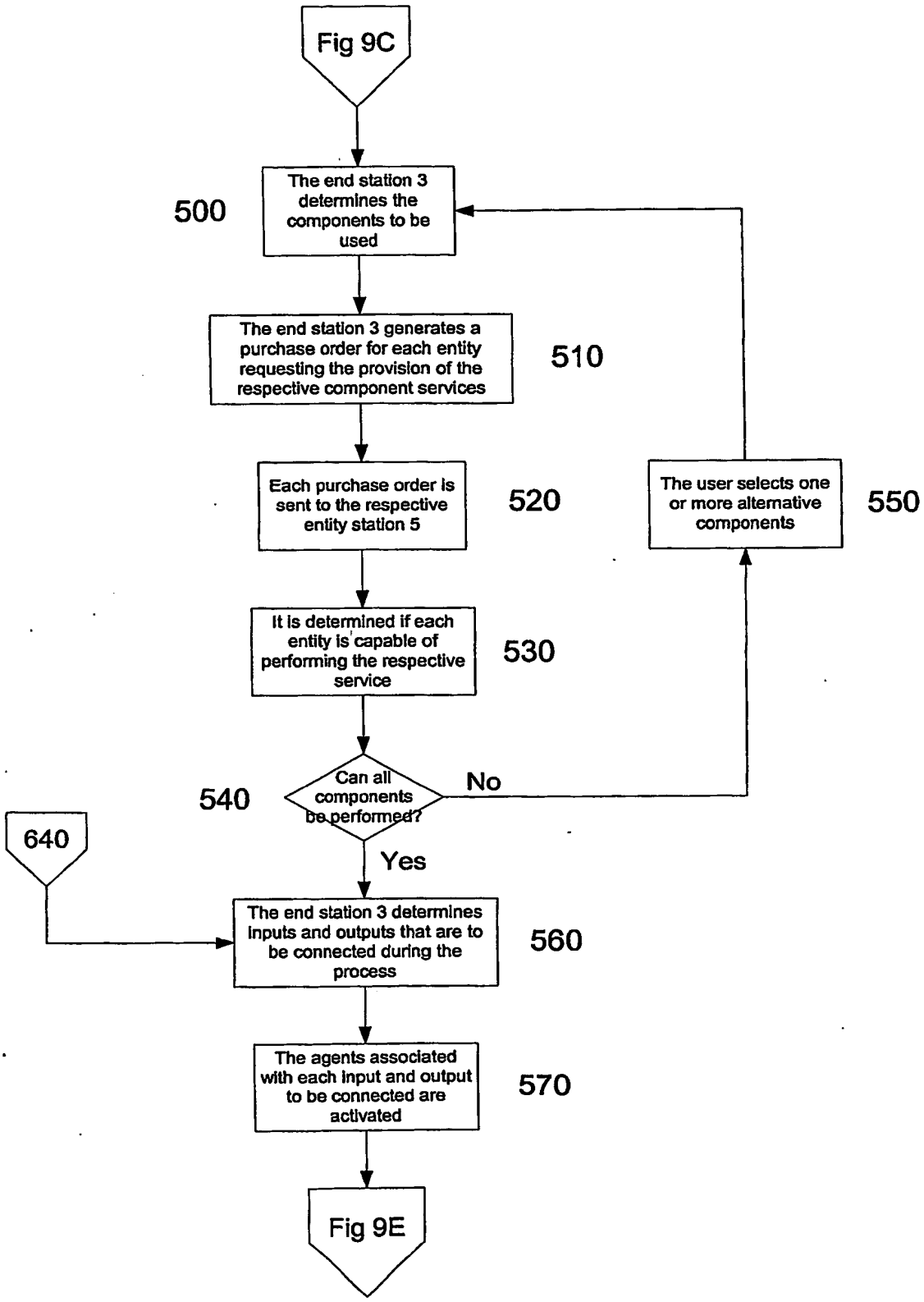
Fig 9C
500
The end station 3 determines the components to be used
The end station 3 generates a purchase order for each entity requesting the provision of the respective component services
510
Each purchase order is sent to the respective entity station 5
520
It is determined if each entity is capable of performing the respective service
530
540
Can all components be performed?
No
The user selects one or more alternative components
550
Yes
640
The end station 3 determines inputs and outputs that are to be connected during the process
560
The agents associated with each input and output to be connected are activated
570
Fig 9E

Fig. 9D

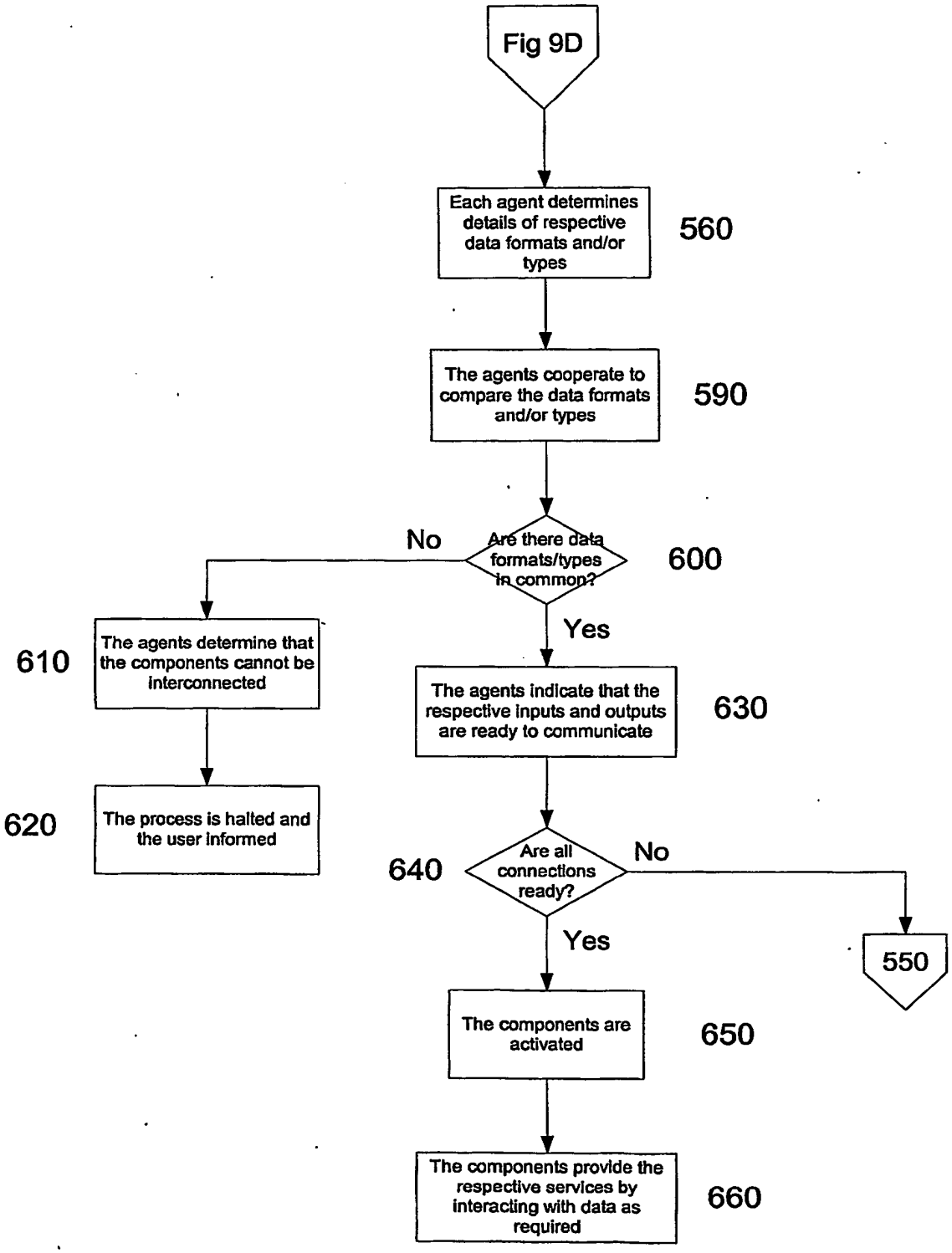
Fig 9D
Each agent determines details of respective data formats and/or types
560
The agents cooperate to compare the data formats and/or types
590
No
Are there data formats/types in common?
600
Yes
610
The agents determine that the components cannot be interconnected
620
The process is halted and the user informed
The agents indicate that the respective inputs and outputs are ready to communicate
630
640
Are all connections ready?
No
Yes
550
The components are activated
650
The components provide the respective services by interacting with data as required
660

Fig. 9E

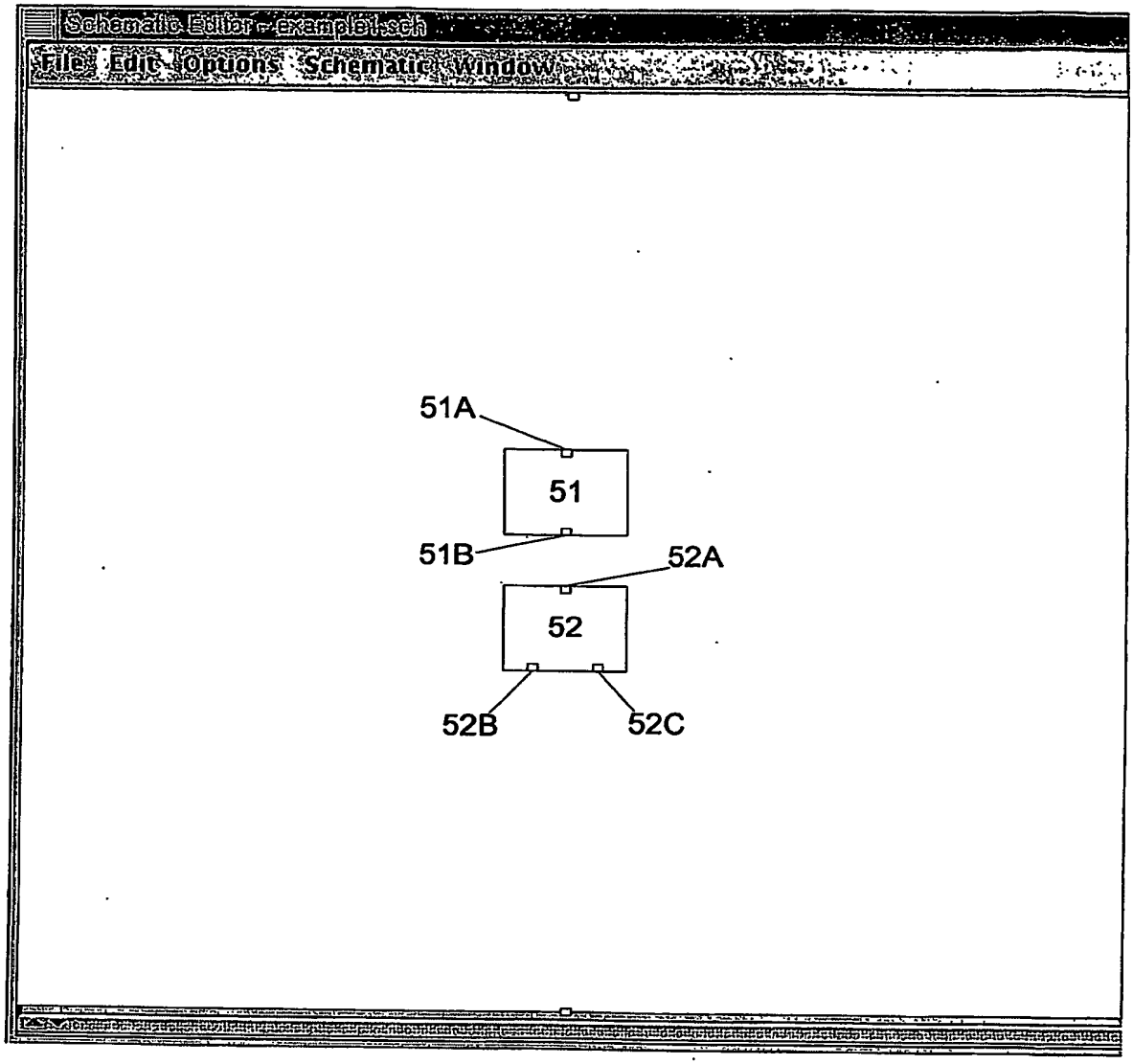
Schematic Editor - example1.sch
File Edit Options Schematic Window
51A
51
51B
52A
52
52B
52C

Fig. 10

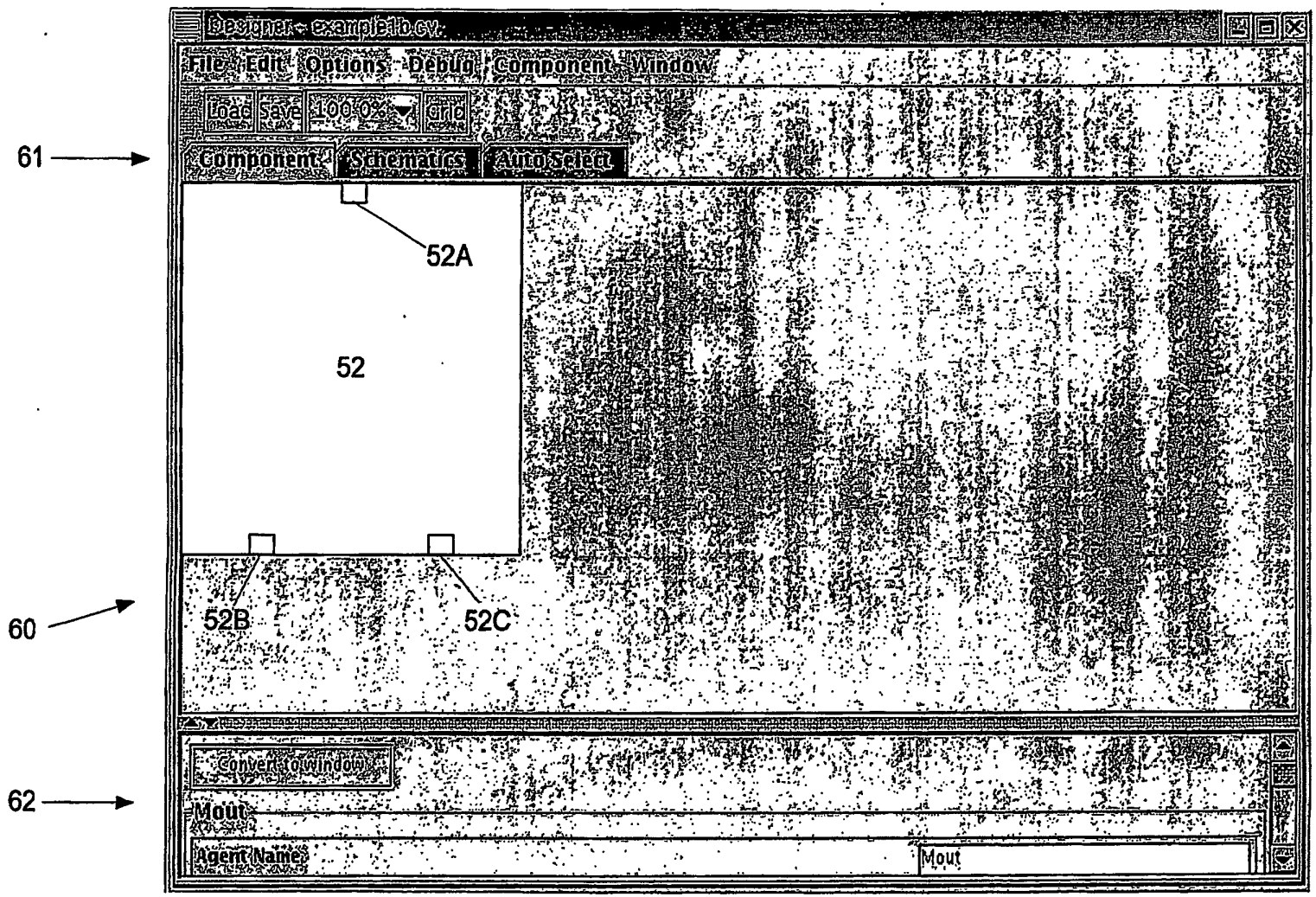
Designer - example1b.cv
File Edit Options Debug Component Window
Load Save 100.0% Grid
Component Schematics Auto Select
52A
52
52B
52C
Convert to window
Mout
Agent Name
Mout
61
60
62

Fig. 11

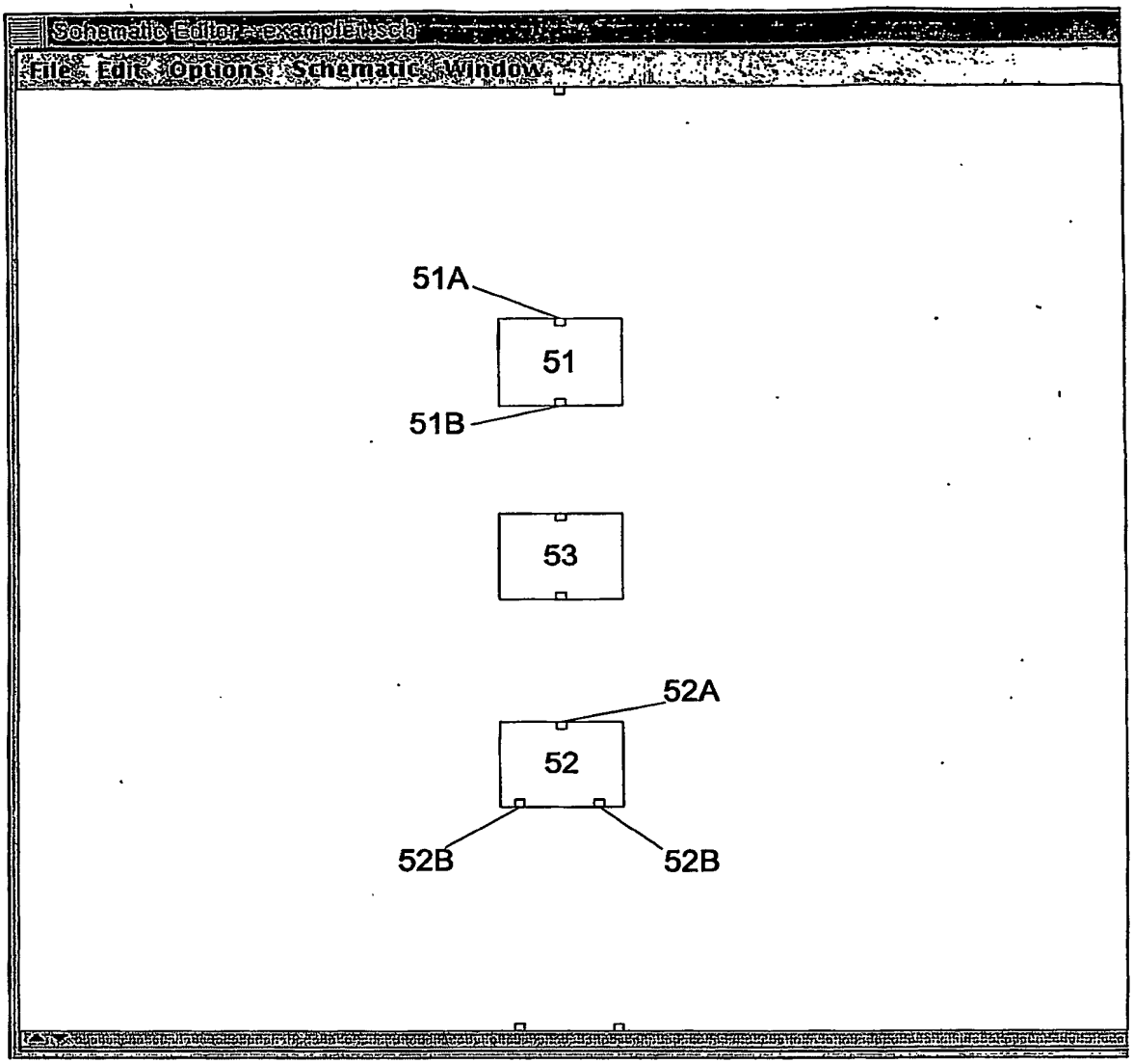
Schematic Editor - example1.sch
File Edit Options Schematic Window
51A
51
51B
53
52A
52
52B
52B

Fig. 12

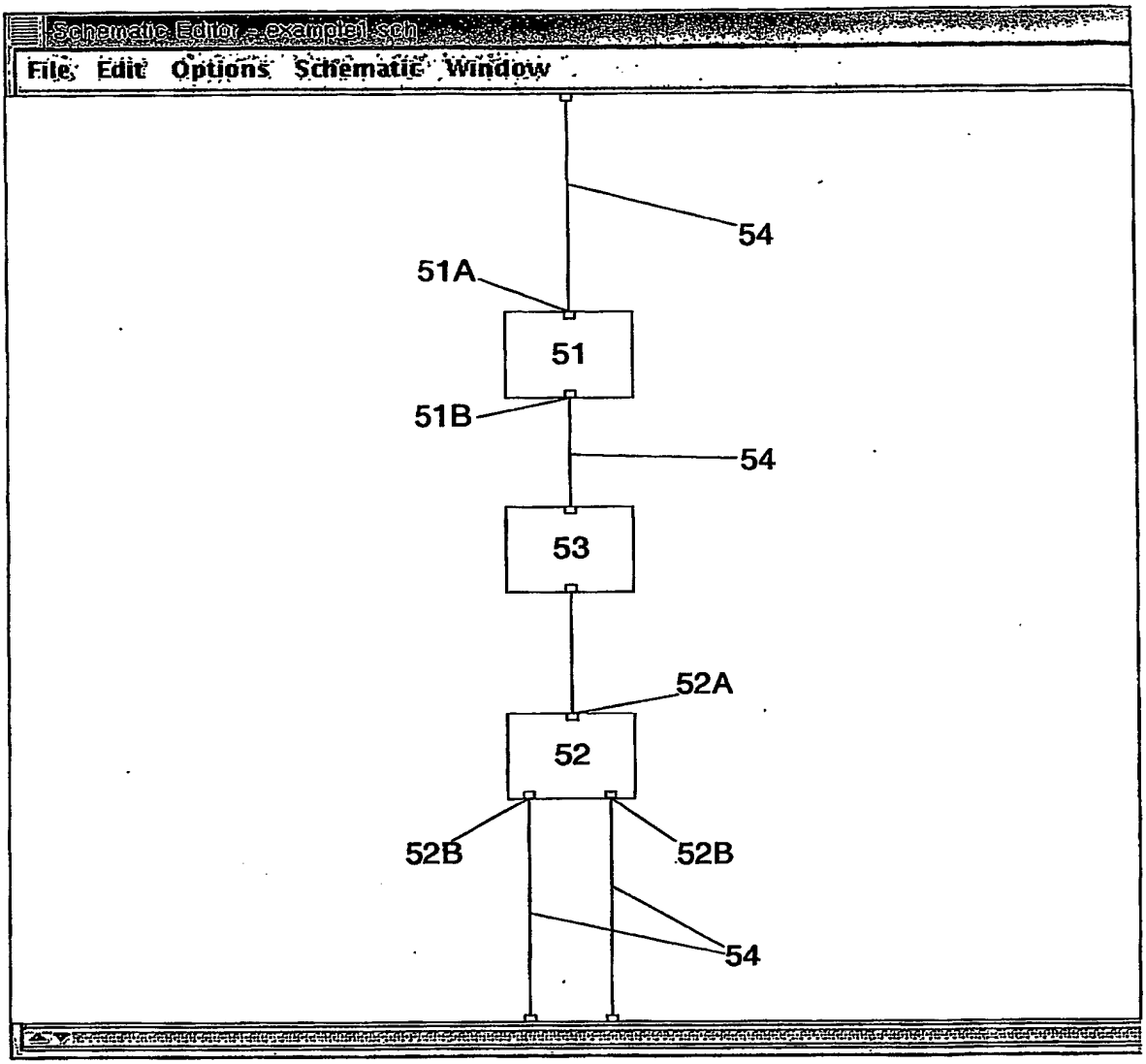
Schematic Editor - example1.sch
File Edit Options Schematic Window
54
51A
51
51B
54
53
52A
52
52B
52B
54

Fig. 13

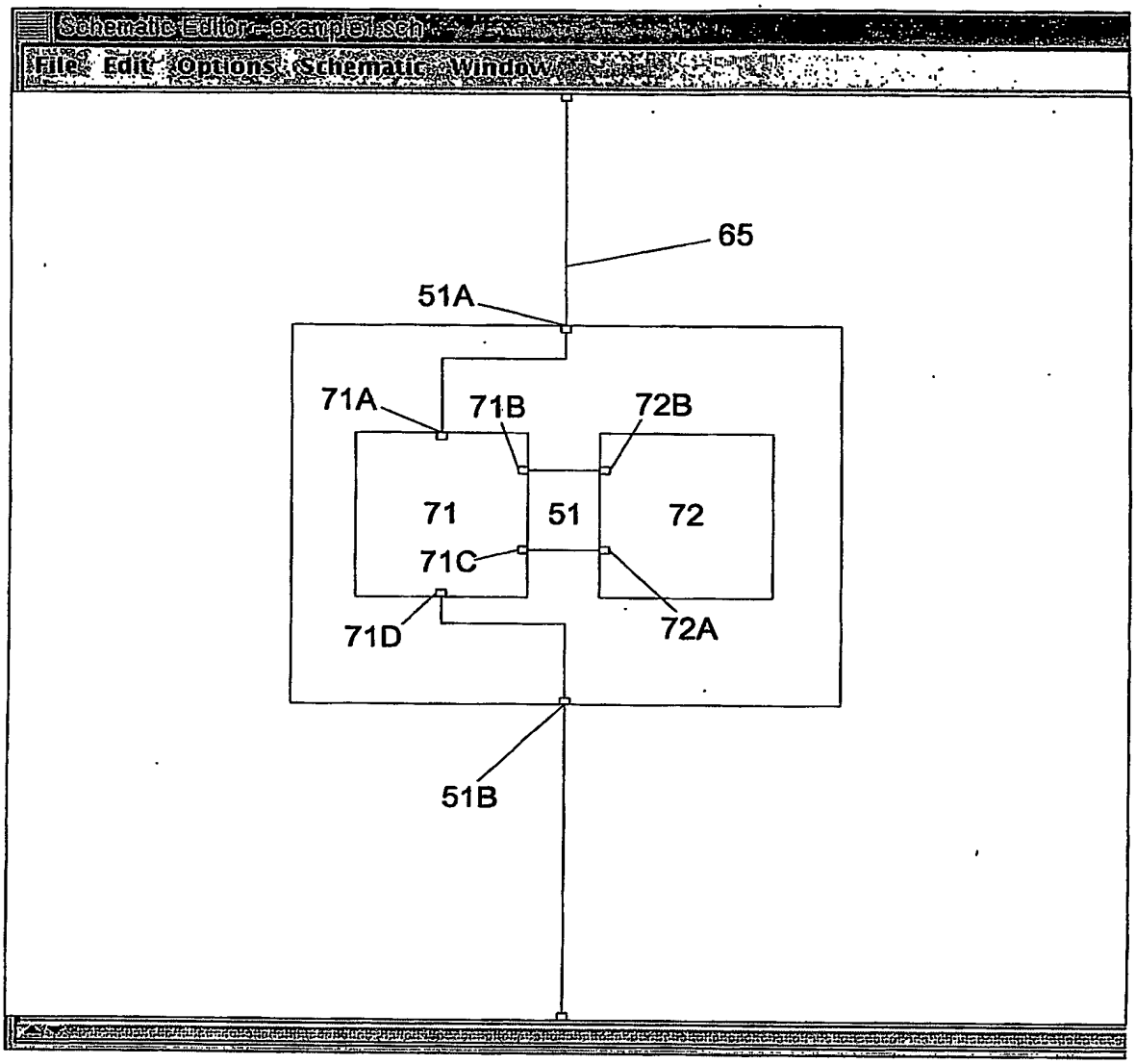
Schematic Editor - example1.sch
File Edit Options Schematic Window
65
51A
71A
71B
72B
71
51
72
71C
71D
72A
51B

Fig. 14

Schematic P

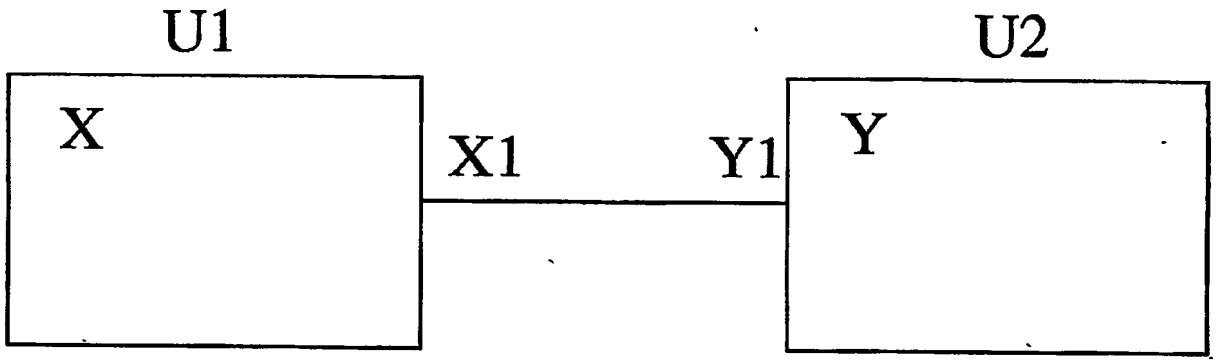


**Fig. 15**

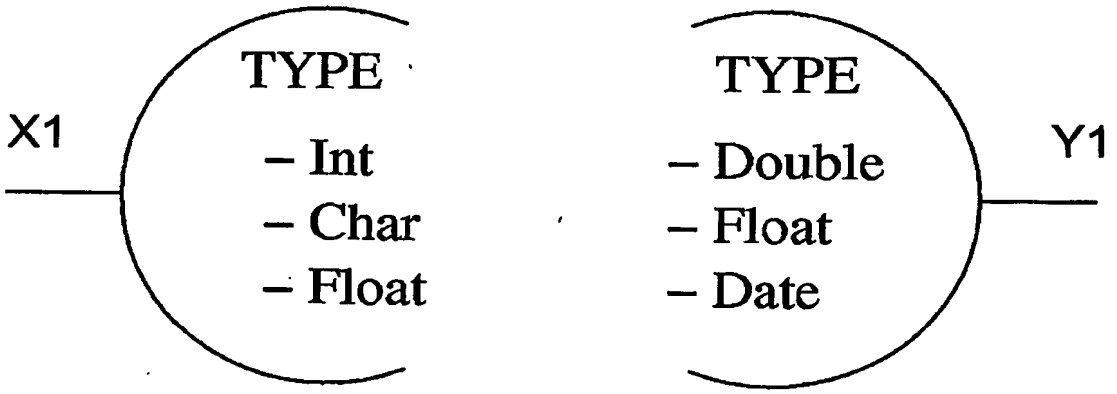


**Fig. 16**

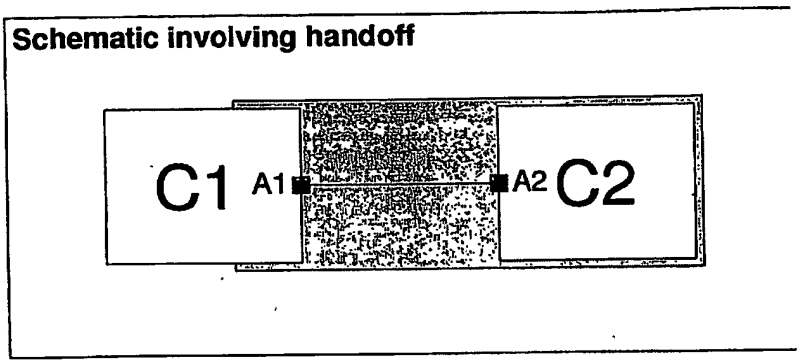


Fig. 17A

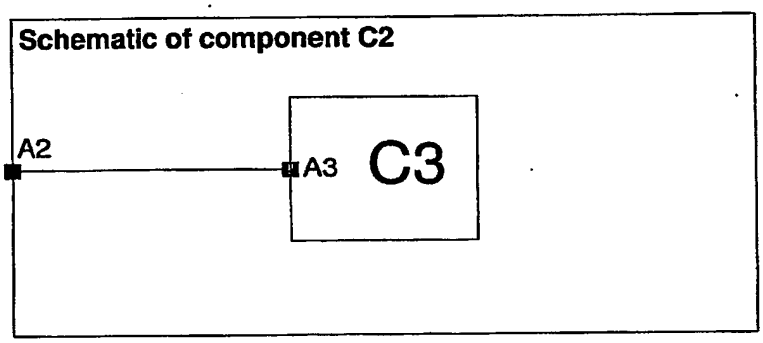


Fig. 17B

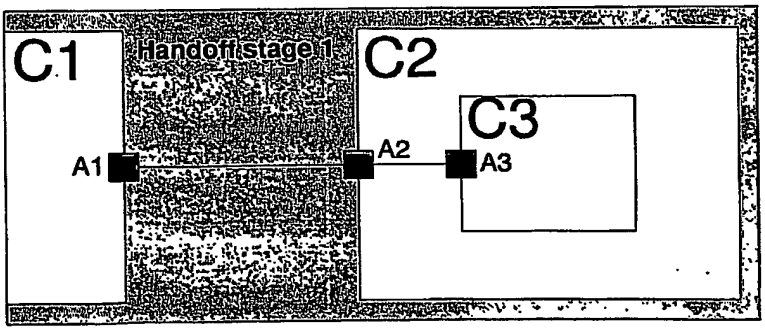


Fig. 17C

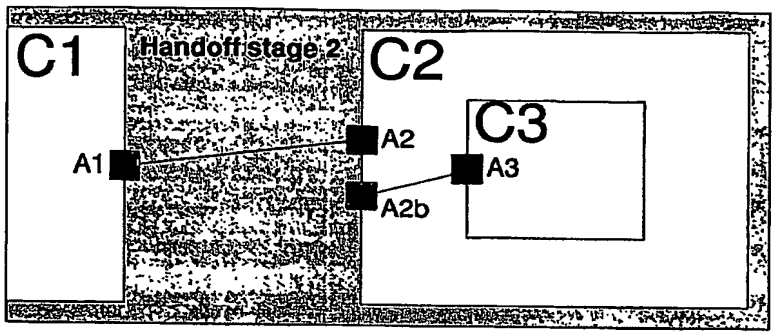


Fig. 17D

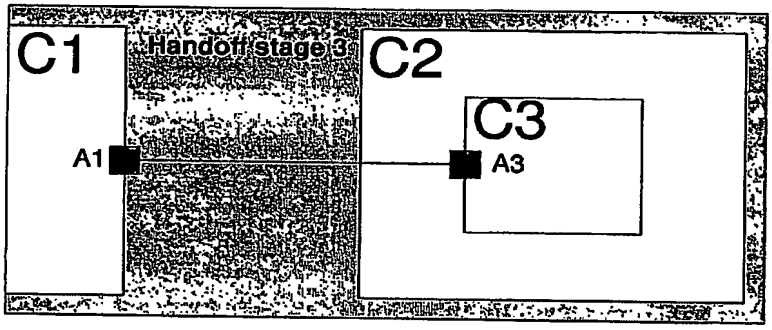


Fig. 17E

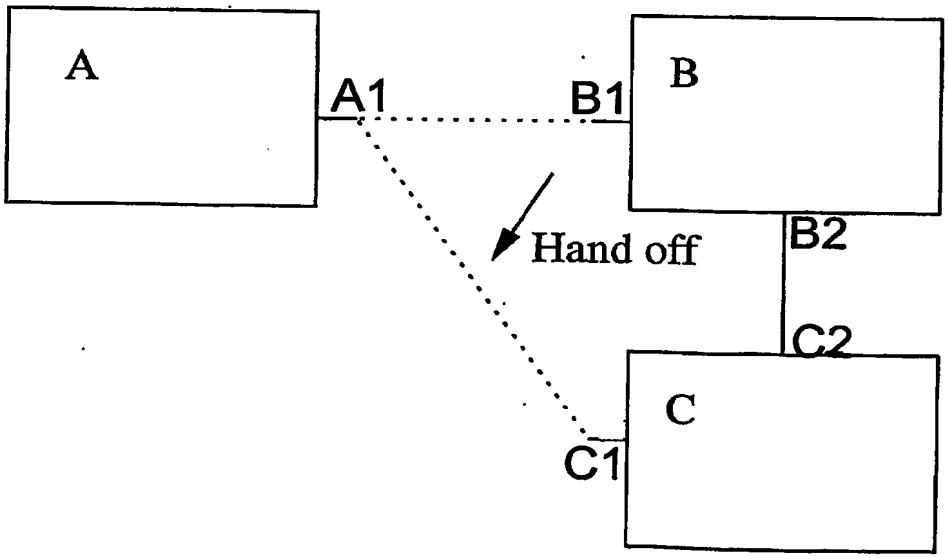


Fig. 18

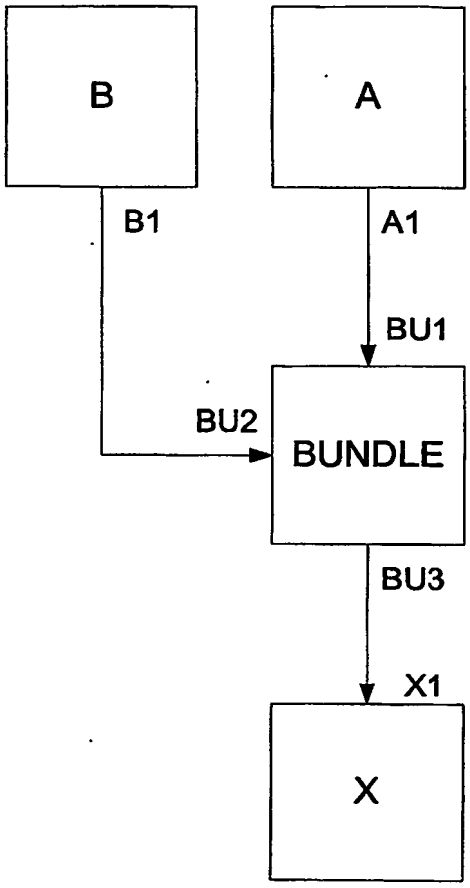


Fig. 19A

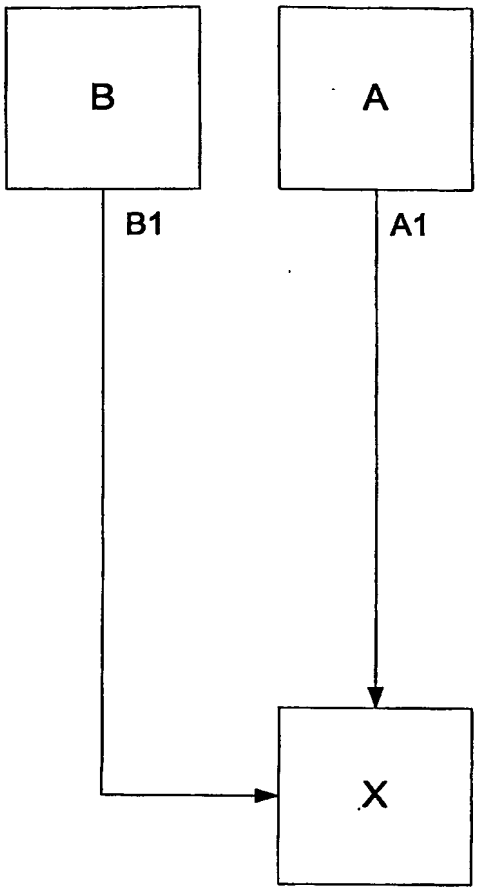


Fig. 19B

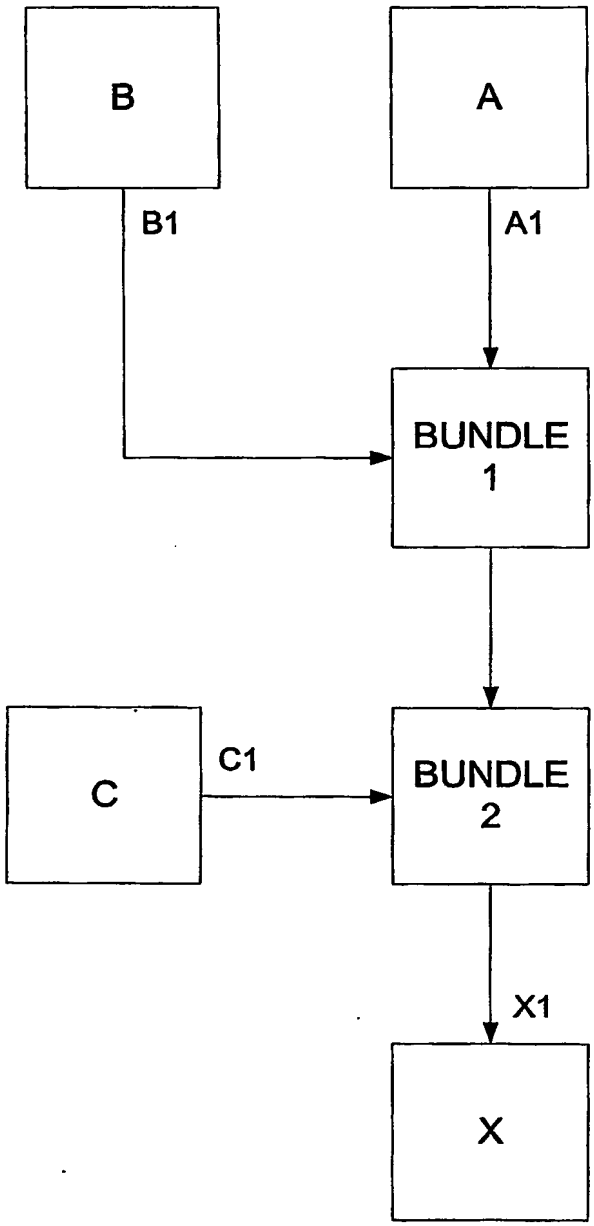


Fig. 20

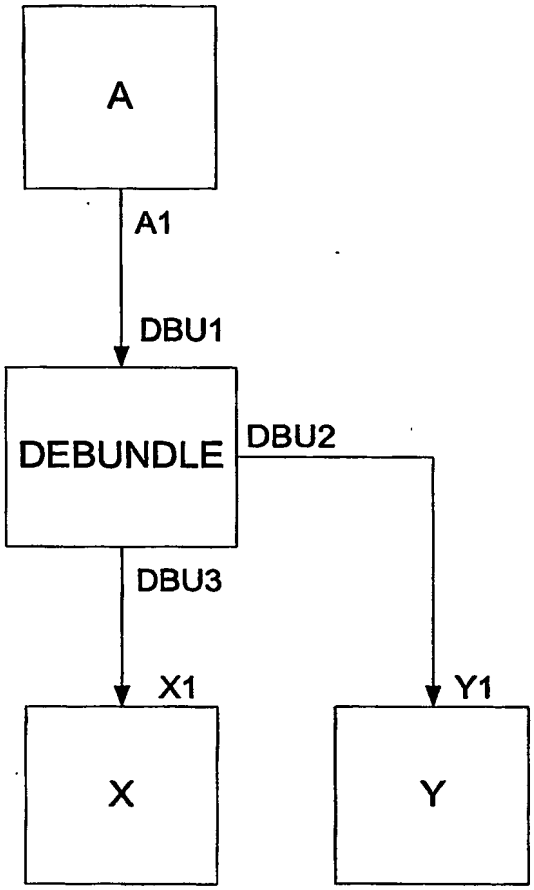


Fig. 21

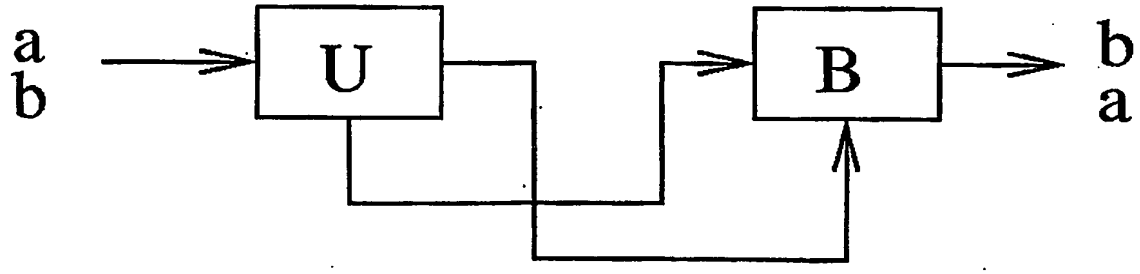


Fig. 23

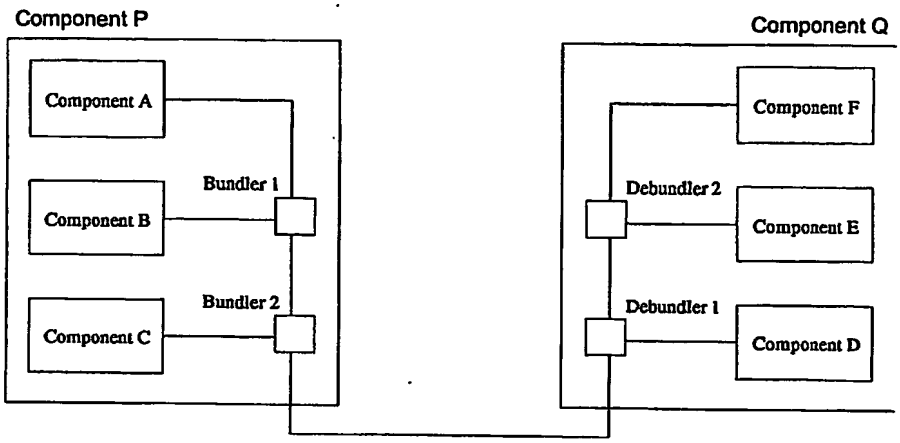


Fig. 22A

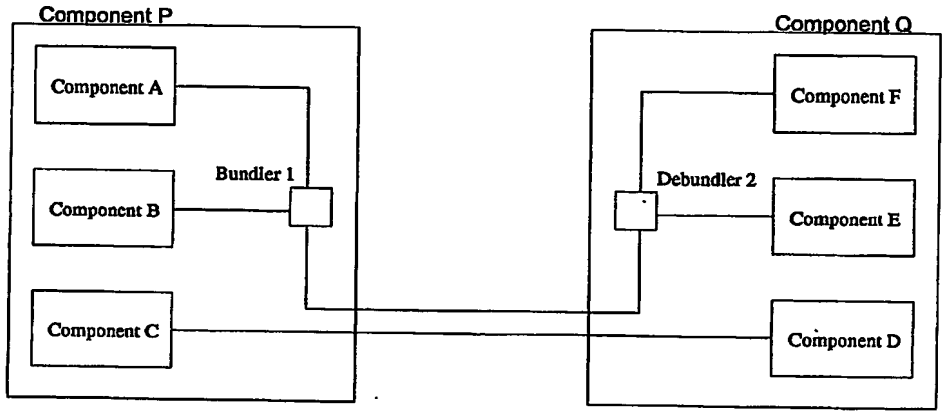


Fig. 22B

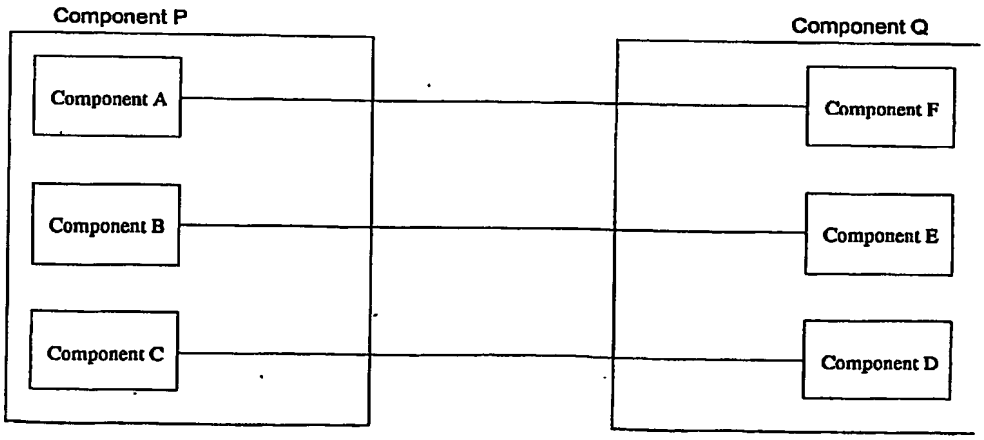


Fig. 22C

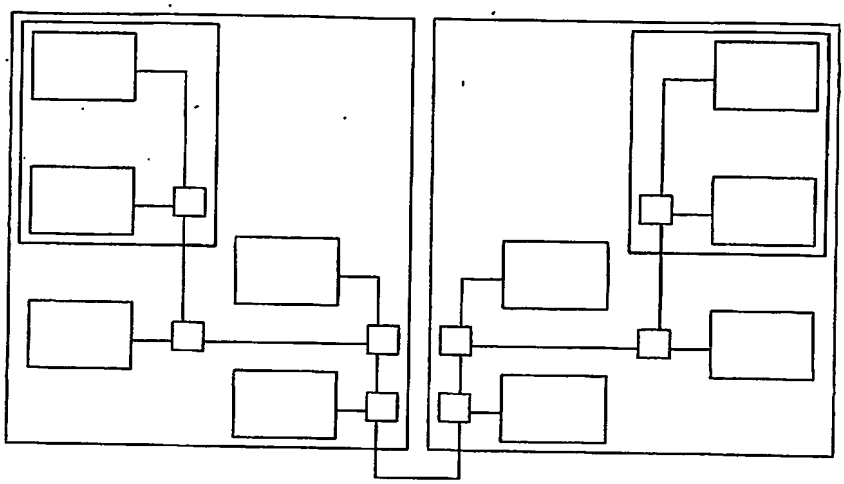

Fig. 22D

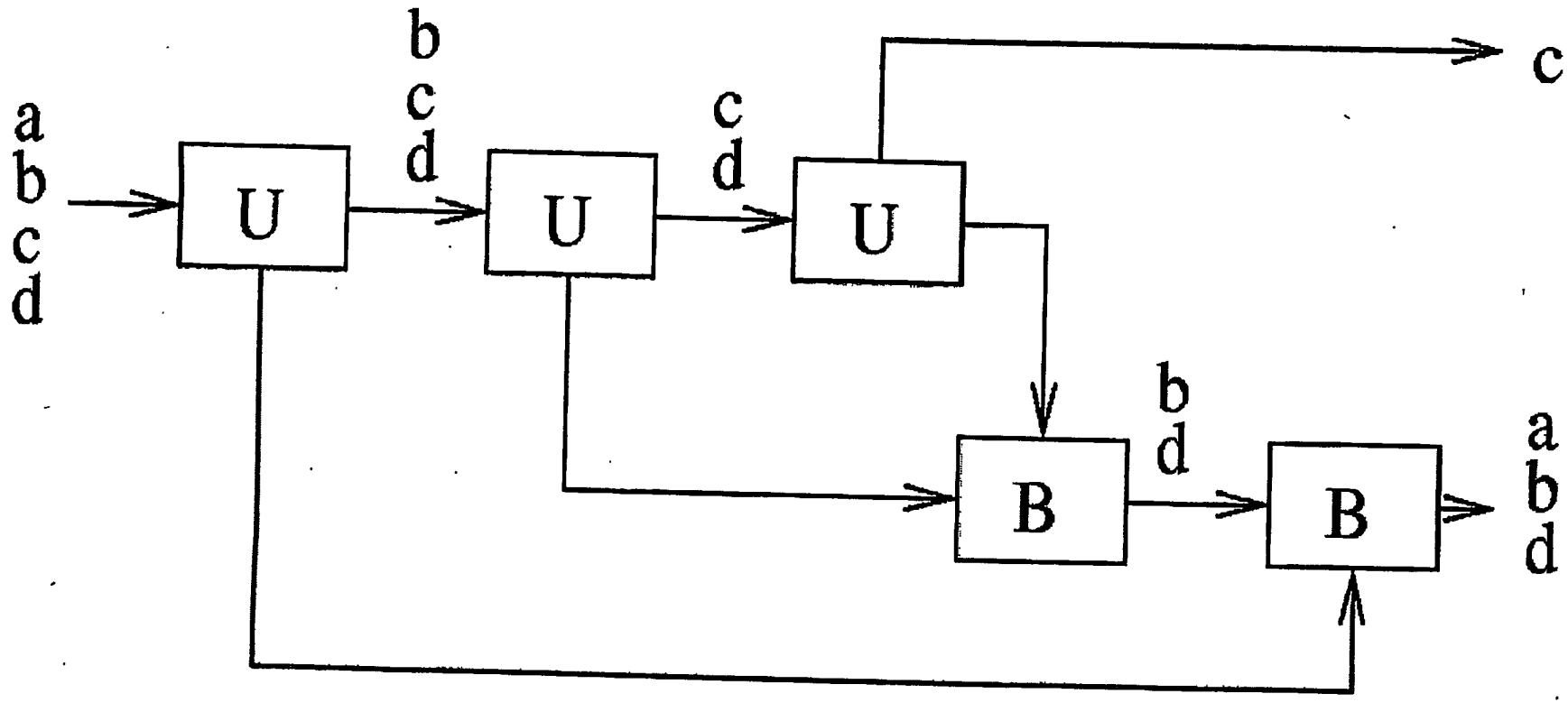


Fig. 24

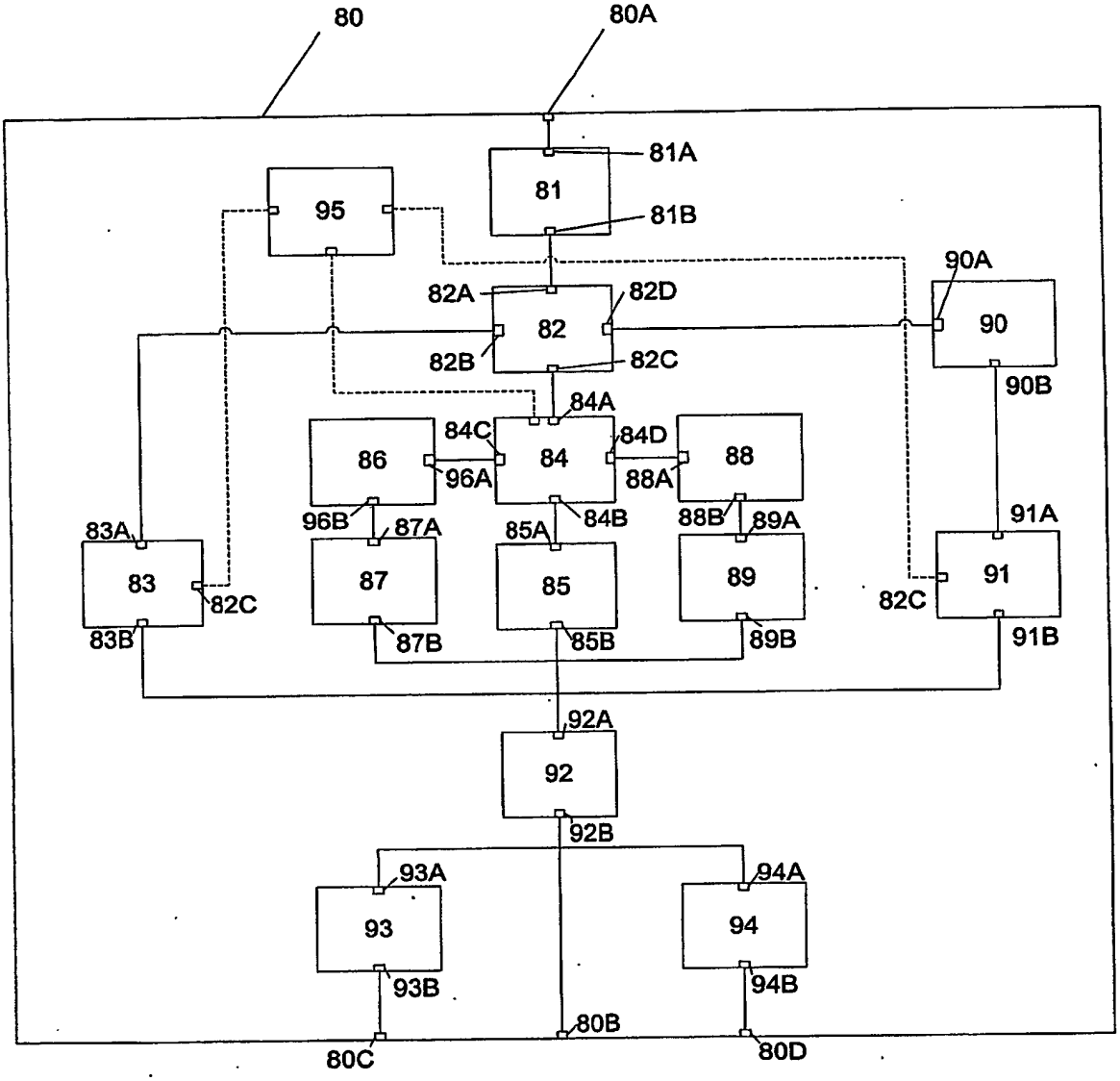
80
80A
81A
81
81B
95
90A
82A
82D
82
90
82B
82C
90B
84A
84C
84D
86
84
88
96A
88A
96B
87A
85A
84B
88B
89A
91A
83A
83
87
85
89
91
82C
82C
83B
87B
85B
89B
91B
92A
92
92B
93A
94A
93
94
93B
94B
80B
80C
80D

Fig. 25

**This Page is Inserted by IFW Indexing and Scanning Operations and is not part of the Official Record**

## BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☐ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

**IMAGES ARE BEST AVAILABLE COPY.**
**As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.**